Date of Filing: August 28, 2003
Japanese Patent Application No: 2003-305513

[Name of Document] Application for Patent
[Reference No.] 0306366
[Date of Filing] August 28, 2003
[Addressee] Commissioner of Japan Patent Office
[Int. Class] G03G 21/00 396
[Inventor]
[Address] c/o Ricoh Company, Ltd.
3-6, Nakamagome 1-chome, Ota-ku, Tokyo
[Name] Hiroyuki Matsushima
[Applicant]
[Id. No.] 000006747
[Address] 3-6, Nakamagome 1-chome, Ota-ku, Tokyo
[Name] Ricoh Company, Ltd.
[Representative] Masamitsu Sakurai
[Intermediate]
[Id. No.] 100080931
[Address] Ikebukuro Whitehouse Building #818,
20-2, Higashi-Ikebukuro 1-chome,
Toshima-ku, Tokyo
[Patent Attorney]
[Name] Hiroshi Osawa
[Priority Document]
[Application No.] 2002-276451
[Date of Filing] September 24, 2002
[Priority Document]
[Application No.] 2002-272978
[Date of Filing] September 19, 2002
[Application Fee]
[Prepayment No.] 014498
[Amount of Payment] 21,000 Yen
[List of Documents Attached]
[Name of Document] Specification 1
[Name of Document] Drawing 1

[Name of Document] Abstract 1

[Generic Authorization No.] 9809113

[Name of Document] Scope of claims

[Claim 1]

A communication method characterized in that:

a first communication apparatus transmits to a second communication apparatus an operation request to be transmitted to the second communication apparatus and an operation response to the operation request received from the second communication apparatus in one batch; and

the second communication apparatus transmits to the first communication apparatus an operation request to be transmitted to the first communication apparatus and an operation response to the operation request received from the first communication apparatus in one batch.

[Claim 2]

The communication method according to claim 1, characterized in that the operation request is a function call and the operation response is an execution result of the function called by the function call.

[Claim 3]

The communication method according to claim 1 or 2, characterized in that transmission of the operation request and the operation response is performed by issuing a communication request always from the second communication apparatus and transmission of the operation request and the operation response from the first communication apparatus is performed as a communication response to the communication request from the second communication apparatus.

[Claim 4]

The communication method according to claim 3, characterized in that the second communication apparatus periodically performs a communication request to the first communication apparatus.

[Claim 5]

A communication method characterized in that:

a first communication apparatus describes a SOAP request to be transmitted to the second communication apparatus and a SOAP response to the SOAP request received from the second communication apparatus in one message and transmits to a second communication apparatus; and

the second communication apparatus a SOAP request to be transmitted to the first communication apparatus and a SOAP response to the SOAP request transmitted from the first communication apparatus describes in one message and transmits to the first communication apparatus.

[Claim 6]

The communication method according to claim 5, characterized in that a function call is described in the SOAP request and an execution result of the function called by the function call is described in the SOAP response.

[Claim 7]

The communication method according to claim 5 or 6, characterized in that:

the second communication apparatus describes a SOAP request and a SOAP response to be transmitted to the first communication apparatus always in an HTTP request and transmits; and

the first communication apparatus describes a SOAP request and a SOAP response to be transmitted to the second communication apparatus always in the HTTP response to the HTTP request and transmits.

[Claim 8]

A communication method according to claim 7, characterized in that the second communication apparatus periodically transmits an HTTP request to the first communication apparatus.

[Name of document] Specifications

[Name of the Invention] Communication Method

[Technical Field]

[0001]

The present invention relates to a communication method in which operation requests and operation responses to the operation requests received from the communication counterparts are transmitted/received among a plurality of communication apparatuses.

[Background Art]

[0002]

Traditionally, in a communication system being connected with a communication apparatus through a network, communication and request are performed to a target communication apparatus by exchanging a message between each other. Then, in such a system, an operation is performed by causing a device to transmit a command as an operation request to another device and an execution result of the operation is returned by the transmitted device as an operation response.

[0003]

The technology as described in above is disclosed in a patent document 1, and in the document, it is described that a remote processor transmits a message for specifying a command to be executed to a local processor and receives a response to the command.

In the document, a technology is also disclosed that when a local processor is arranged inside of a firewall, a command may be transmitted from the outside of the firewall to the inside by transmitting a communication request from the local processor to a remote process at the outside of the firewall and transmitting a command by the remote processor to the local processor as a response to the communication request.

[Patent document 1] Japanese Patent Laid Open Publication No. 2001-273211

[0004]

Technologies related to such operation requests may be applied to a system for remote controlling an operation of a device connected

with a communication apparatus. In the patent document 2, an example is described that such technology is applied to a remote operation device system for causing a remote target operation device having an operation function of a blind and a light to operate a blind and a light by transmitting a command from a remote operation device having a function for accepting an operation from a user. However, transmitting a response to the command is not mentioned in the document.

[Patent document 2] Japanese Patent Laid Open Publication No. 2002-135858

[Disclosure of the Invention]

[Problems to be Solved by the Invention]

[0005]

When messages are exchanged among a plurality of communication apparatuses, a communication apparatus to send commands are not necessarily limited to one. Alternatively, a plurality of communication apparatuses may be arranged to send commands to each other, in which case the communication apparatus receiving a command is requested to send back an execution result to the sender of the command. In such an arrangement, information sent from one communication apparatus to a counterpart communication apparatus may be a command to the counterpart communication apparatus or an execution result of a command received from the counterpart communication apparatus.

[0006]

In the conventional art, these commands and execution results are sent separately. In such a method, separate connections are respectively established for the time a command is to be sent and for the time an execution result of the received command is to be sent. In turn, the communication overhead is increased and thereby a problem in communication efficiency arises.

Presently, there are still many environments implementing the dial-up connection for establishing connection via a network. The

above-described problem is particularly troublesome in such environments. Namely, in such environments, it may take several tens of seconds to establish a connection, and a fee is charged each time a connection is established. Therefore, an increase in the number of connections established results in a significant rise in costs.

The present invention is for solving the above-described problems and its object is to improve communication efficiency for the transmission/reception of an operation request and an operation response to the operation request each other among a plurality of communication apparatuses.

[Means for solving the problems]

[0007]

For achieving the above objectives, when communicating between a fist communication apparatus and a second communication apparatus, the communication method of the present invention is configured such that the first communication apparatus transmits to a second communication apparatus an operation request to be transmitted to the second communication apparatus and an operation response to the operation request received from the second communication apparatus in one batch, and the second communication apparatus transmits to the first communication apparatus an operation request to be transmitted to the first communication apparatus and an operation response to the operation request received from the first communication apparatus in one batch.

[0008]

In the communication method according to the above, the operation request may be a function call and the operation response may be an execution result of the function called by the function call.

Further, transmission of the operation request and the operation response may be performed by issuing a communication request always from the second communication apparatus and transmission of the operation request and the operation response

from the first communication apparatus may be performed as a communication response to the communication request from the second communication apparatus.

Further more, the second communication apparatus may periodically perform a communication request to the first communication apparatus.

[0009]

Also, the present invention provides a communication method in which a first communication apparatus describes a SOAP request to be transmitted to the second communication apparatus and a SOAP response to the SOAP request received from the second communication apparatus in one message and transmits to a second communication apparatus, and the second communication apparatus a SOAP request to be transmitted to the first communication apparatus and a SOAP response to the SOAP request transmitted from the first communication apparatus describes in one message and transmits to the first communication apparatus.

[0010]

In the communication method according to the above, a function call may be described in the SOAP request and an execution result of the function called by the function call may be described in the SOAP response.

Further, the second communication apparatus may describe a SOAP request and a SOAP response to be transmitted to the first communication apparatus always in an HTTP request and transmits, and the first communication apparatus may describe a SOAP request and a SOAP response to be transmitted to the second communication apparatus always in the HTTP response to the HTTP request and transmits.

Further more, the second communication apparatus may periodically transmit an HTTP request to the first communication apparatus.

[Effect of the Invention]

[0011]

According to the communication method of the present invention as described in above, since an operation request to a communication counterpart and an operation response to an operation request received from a communication counterpart may be transmitted only by establishing a connection once, overhead may be reduced and communication efficiency may be improved.

[Best Mode for Carrying out the Invention]

[0012]

In the following, a best embodiment of the invention is described referring to drawings.

Fig. 1 shows an exemplary configuration of a communication system for using the present invention.

The communication system is configured by connecting a first communication apparatus 1 and a second communication apparatus 2 by a network 10 as shown in Fig. 1.

Then, the first communication apparatus 1 and the second communication apparatus 2 may be configured as a various type electronic apparatus including a communication function and an information process function as well as a computer such as a PC including communication function. As for the network 10, the Internet, a LAN (local area network), or other various communication routes that enable network communication may be used regardless of whether it is wired or wireless.

[0013]

The first communication apparatus 1 and the second communication apparatus 2 are mounted with various application programs for controlling and managing each other. Then, each node is arranged to send an 'operation request', which corresponds to a request for an execution of a method (process) of an application program implemented in each node, and acquire an 'operation response', which corresponds to a processing result of the requested process, using RPC (remote procedure call). In other words, the first communication apparatus

1 is capable of generating a request to the second communication apparatus 2 (referred to as first communication apparatus request, hereinafter), transmitting the first communication apparatus request to the second communication apparatus 2, and acquiring a response to this request. The second communication apparatus 2 is capable of generating a request to the first communication apparatus 1 (referred to as second communication apparatus request, hereinafter), transmitting the second communication apparatus request to the first communication apparatus 1, and acquiring a response to this request.

It is noted that in the present application, a method is defined as a logical function prescribing an input and output format. Thus, the operation request corresponds to a function call (Procedure Call) for calling the function, and the operation response corresponds to the execution result of the function called by the procedure call.

[0014]

Fig.2 shows a relation between the operation request and the operation response.

Fig. 2(A) illustrates a case in which an operation request to the second communication apparatus 2 is generated at the first communication apparatus 1. In this model, the first communication apparatus 1 generates a first communication apparatus operation request and sends this to the second communication apparatus 2, and the second communication apparatus 2 receiving this request sends back an operation response to this request.

[0015]

Fig. 2(B) illustrates a case in which an operation request to the first communication apparatus 1 is generated at the second communication apparatus 2. In this model, the second communication apparatus 2 generates a second communication apparatus operation request and sends this to the first communication apparatus 1, and the first communication apparatus 1 receiving this request sends back an operation response to this request.

It is noted that in the present embodiment, SOAP (simple object

access protocol) is used as a protocol for transmitting the argument and return value by an RPC, and the above-described operation request and operation response are referred to as SOAP messages.

[0016]

The present invention is characterized in that, in an arrangement where a plurality of communication apparatuses transmit/receive operation requests and operation responses to received operation requests to/from each other, as described above, for example, an operation request that is to be sent to a counterpart communication apparatus and an operation response to an operation request received from the counterpart communication apparatus are combined and collectively sent.

In the present invention, as a communication protocol for transmitting an operation request or an operation response, for example, HTTP (HyperText transfer Protocol) or SMTP (Simple Mail Transfer Protocol) may be used as an appropriate communication protocol depending on the configuration of the communication system. However, since no corresponding relation such as a communication request and a communication response is involved in an electronic email to be sent using SMTP, a communication apparatus using SMTP is not included in the embodiment of the communication client of the invention.

Thus, first regarding the communication method of the present invention, an embodiment using HTTP as the communication protocol is described, and then a case of using SMTP as the communication protocol is described as a reference example.

[0017]

[Embodiment Using HTTP: Figs. 3 to 21]

Fig. 3 shows an exemplary configuration of a communication system applying an embodiment using HTTP. As is shown in Fig. 3, this communication system includes an HTTP server 12, an HTTP client 11, and Internet 13 that interconnects the HTTP server 12 and the HTTP client 11. However, the HTTP client 11 is connected to the Internet

13 via a firewall 14 to ensure security. The HTTP server 12 corresponds to the first communication apparatus, and the HTTP client 11 corresponds to the second communication apparatus.

[0018]

In performing communication using HTTP, a node stationed within the firewall 14 may not be freely accessed from outside of the firewall, and data is sent to the node only as a communication response (HTTP response) to a communication request (HTTP request) sent from this node. In this communication system, the HTTP client 11 corresponds to the node stationed within the firewall 14 and the HTTP server 12 corresponds to the node stationed outside the firewall 14. Therefore, in implementations other than the communication between the above nodes, these nodes are not limited to functioning as a server or a client.

[0019]

Also, as same as in the first communication apparatus 1 and the second communication apparatus 2 shown in Fig. 1, the HTTP server 12 and the HTTP client 11 include application programs to control and manage each other. Further, using RPC (remote procedure call), the HTTP server 12 and the HTTP client 11 are capable of sending to each other an 'operation request' corresponding to a call to perform a method (process) of an application program implemented at the other side, and acquiring an 'operation response' corresponding to the processing result of this 'operation request'.

[0020]

Fig. 4 illustrates a relation between the operation requests and the operation responses.

Fig. 4(A) illustrates a case in which an operation request to the HTTP server 12 is generated at the HTTP client 11. In this model, the HTTP client 11 generates a client operation request (hereinafter, referred to as 'client command') and sends this request to the HTTP server 12, and the HTTP server 12 receiving this request sends back an operation response to the command (also, hereinafter, referred to

as 'command response' or simply 'response').

[0021]

Fig. 4(B) illustrates a case in which an operation request to the HTTP client 11 is generated at the HTTP server 12. In this model, the HTTP server 12 generates a server operation request (hereinafter, referred to as 'server command') and sends this request to the HTTP client 11, and the HTTP client 11 receiving this command sends back an operation response to this command.

As is described above, in the RPC level, the operation request and the operation response are equally handled between the HTTP client 11 and the HTTP server 12. However, in the communication level, both are not equally handled.

[0022]

Fig. 5 illustrates an exemplary communication sequence in the communication system according to the present embodiment.

As is shown in the drawing, in this communication system, the HTTP client 11 sends an HTTP request to the HTTP server 12 as a communication request, and the HTTP server 12 sends back an HTTP response to the HTTP client 11 as a communication response to the communication request. For example, the HTTP client 11 may send an HTTP request X and the HTTP server 12 may respond by sending back an HTTP response X, or similarly, the HTTP client 11 may send an HTTP request Y and the HTTP server 12 may respond by sending back an HTTP response Y.

[0023]

In this embodiment, an HTTP request of a communication sequence may include and sent, a client command corresponding to an operation request to be sent from the HTTP client 11 to the HTTP server 12 and a response to a server command that has been sent from the HTTP server 12 to the HTTP client 11 (command response). Similarly, an HTTP response may include and sent, a server command corresponding to an operation request to be sent from the HTTP server 12 to the HTTP client 11 and a response to a client command that has been sent from the HTTP client 11 to the HTTP server 12 (command response).

[0024]

Therefore, for example, a client command A may be described in the HTTP request X and transferred, and a corresponding command response may be described in the HTTP response X corresponding to the HTTP request X and transferred. However, a server command C is described in the HTTP response X corresponding to the HTTP request X and transferred, and a corresponding command response to the server command C is described in an HTTP request Y and transferred.

[0025]

Also, in the case of Fig. 4(A), the HTTP client 11 is able to establish a connection with the HTTP server 12 immediately after a client command is generated to include the generated command in an HTTP request and transmit this to the HTTP server 12. In the case of Fig. 4(B), the firewall 14 implemented at the HTTP client 11 side blocks an HTTP request from the HTTP server 12 side and, therefore, the HTTP server 12 is unable to establish a connection with the HTTP client 11 to transmit a generated server command right away.

[0026]

It is also noted that any number (including 0) of client commands and responses to server commands may be described in one HTTP request, and any number (including 0) of server commands and responses to client commands may be described in one HTTP response. The contents described in one HTTP request or HTTP response are logically transferred as one bundle.

In this way, the number of connections required in transmitting information may be reduced so that communication overhead is reduced and communication efficiency is enhanced.

[0027]

Fig. 6 illustrates another exemplary communication sequence in the communication system according to the present embodiment.

In Fig. 5, a very simple sequence example has been given in order to simplify the explanation. In Fig. 6, an example is given in which the number of commands and command responses described in each HTTP

request or HTTP response is varied.

When a command is received, a response to the command may not be sent back at the next transmission opportunity. For example, as a client command B shown in Fig. 6, the corresponding command response to the client command B may not be described in the HTTP response X′ of the HTTP request X′ and send back. Instead, the command response may be described in a subsequent HTTP response Y′ and send back.

Of course, the above may apply to the server command as well. Thus, a command response to a server command described in an HTTP response may not be described in the subsequent HTTP request. The response may be described in any further HPPT request that is transmitted after the receipt of this HTTP response and transferred.

[0028]

It is noted that each command and command response is generated independently and implemented for process, thus, when one or more commands and command responses are to be processed together to be transferred in one bundle as described above, these commands and command responses are necessary to be combined before the transmission and to be separated after the transmission. In the following, a hardware configuration of the HTTP client 11 and the HTTP server 12, a functional configuration for realizing the above described processes, and procedural steps for realizing the processes are described.

[0029]

Fig. 7 shows an exemplary hardware configuration of the HTTP client 11 and HTTP server 12.

As is shown in the drawing, each HTTP client 11 and HTTP server 12 includes a CPU 31, a ROM 32, a RAM 33, an SD (Secure Digital) card 34, a network interface card (NIC) 35, wherein a system bus 36 interconnects these elements.

[0030]

Further, these components are described. The CPU 31 is a control means for realizing overall control of the HTTP client 11 or HTTP server 12 entirely using a control program stored in the ROM 32. The ROM

32 is a read only memory storing various kinds of fixed data including the control program used by the CPU 31.

The RAM 33 is a temporary storage memory used as a working memory for the CPU 31 to perform data processing. The SD card 34 is a nonvolatile memory that is able to retain its stored contents even when the power of the apparatus is turned off. The NIC 35 is a communication means for transmitting/receiving information to/from a communication counterpart via a network such as the Internet 13.

[0031]

Fig. 8 is a block diagram illustrating a functional configuration of the HTTP client 11 for realizing processes pertaining to commands and command responses.

Of the functions shown in Fig. 8, a client command pool 41 and a server command pool 42 are arranged to any rewritable storage means. For example, these command pools may be arranged to the SD card 34, or may be arranged to the RAM 33 and to an HDD (hard disk drive) not shown. A client command generating means 43, a server command execution result generating means 44, a transmission message collection means 45, and a received message distributing means 48 are realized by the CPU 31. Also, HTTP request transmission means 46 and HTTP response receiving means 47 are realized by the CPU 31 and the NIC 35.

[0032]

In the following, further details of the above functions are described.

The client command pool 41 is a pool for registering a client command, a response to the command, and identification information to the command in association with each other. Also, the server command pool 42 is a pool for registering a server command, a response to the command, and identification information to the command in association with each other.

The client command generation means 43 generates a client command, assigns identification information (ID) for identifying the command,

attaches the management information for managing the command, and registering in the client command pool 41 these information in association with each other as a client command sheet having a table format. Among the information, the part where a client command is generated corresponds to an application provided, for example, in the HTTP client 11. Also, the client command generation means 43 may provide the generated client command with a function for attaching a priority order for executing each command to the HTTP 12.

[0033]

Herein, an example of a data configuration in a client command sheet is shown in Fig. 9. As shown in Fig. 9, in the client command sheet, an area for storing data of 'command ID', 'method name', 'input parameter', 'status', 'client command execution result notifying destination' and 'output parameter' is provided. Then, among the information, 'command ID', 'method name', and 'input parameter' correspond to the client command (and ID attached there), and 'status' and 'client command execution result notifying destination' correspond to the management information. The 'output parameter' corresponds to the content of the command response received from the HTTP server 12.

[0034]

In the following, a specific description of each of the above items is given.

First, the 'method name' corresponds to a content of a request being made to the HTTP server 12, and indicates a type of function to be called at the HTTP server 12. The 'input parameter' corresponds to data attached to the 'method name', and indicates an argument for calling a function. The 'command ID' corresponds to identification information for identifying a client command. The 'status' corresponds to data indicating a progress of a process relating to the client command, and changes from 'not transmitted' → 'waiting for response' → 'response received' according to the progress of the process.

[0035]

The 'client command execution result notification destination' is reference information indicating a module to be notified for causing a necessary process to be executed, when a response to a client command described in a client command sheet is received. The module being referred to usually corresponds to an application program that has generated the client command. However, this is not necessarily the case. In the 'output parameter', a content of a command response is stored at the time the command response is received. The 'output parameter' is left blank until the command response is received from the HTTP server 12.

[0036]

Referring back to Fig. 8, the server command execution result generating means 44 corresponds to an application for reading from the server command pool 42 and executing a server command. A server command received from the HTTP server 12 is registered in the server command pool 42 as a server command sheet having a table format in association with an ID for identifying the command and management information for managing the command. Thus, the command response generated by the server command execution result generating means 44 is also registered in the server command sheet of the corresponding server command executed.

[0037]

The server command execution result generating means 44 may be provided with a function of reading a plurality kind of server commands from the server command pool 42 and generating a response to each server command. In the case where the server commands include priority information causing the HTTP client 11 to prioritize the execution of their respective processes, the server command execution result generating means 44 may be provided with a function of reading and executing the respective processes according to the server command having the highest priority.

Also, it is noted that the server command execution result

generating means 44 may correspond to, not an application program, but a module that administers the execution of the command by calling the appropriate application program for executing the server command.

[0038]

Fig. 10 shows an exemplary data configuration of a server command sheet.

As is shown in Fig. 10, the server command sheet includes areas for storing data corresponding to a 'command ID', a 'method name', an 'input parameter', a 'status', an 'output parameter', and a 'server command notifying destination'. The 'command ID', the 'method name', and the 'input parameter' correspond to the server command (and ID attached thereto), the 'status' and the 'server command notifying destination' correspond to the management information. The 'output parameter' corresponds to the server command execution result, which indicates the content of the command response that the HTTP client 11 sends back in response to the server command.

[0039]

In the following, the content of each data of the above is described.

First, the 'method name' corresponds to the content of a request being made to the HTTP client 11, and indicates the type of function being called in the HTTP client 11. The 'input parameter' corresponds to data attached to the 'method name', and indicates the argument for calling a function. The 'command ID' corresponds to information for identifying a server command. The 'status' corresponds to data indicating the status of a process pertaining to a server command, and changes from 'not processed' → 'process complete' → 'response made' according to the progress of the process. The 'output parameter' stores a response generated by the server command execution result generating means 44. This item is left blank until the execution of the server command is completed and the 'status' reaches 'process complete'. The 'server command notifying destination' corresponds to reference information indicating a module for executing a server command.

[0040]

Referring back to Fig. 8, the transmitting message collection means 45 has a function for reading from the server command pool 42 a command response generated by the server command execution result generating means 44 in association with the command ID of the server command corresponding to this command response, as well as reading from the client command pool 41 a client command generated by the client command generating means 43 in association with the command ID of this command, and generating an transmitting message from the read data.

When execution priority order is assigned to the command responses and/or the client commands, the transmission message collection means 45 may be arranged to read the command responses and/or client commands according to the order of execution priority.

[0041]

Herein, a transmission message corresponds to the command response or the command and the corresponding command ID being described as a SOAP message using the XML (Extensible Markup Language), which is a structured language format. Then, the transmission message collection means 45 generates a SOAP message as a transmission message for each one of command responses or commands. In such case, the corresponding command ID of each command is described in a SOAP header and the command response or the content of the client command is described in a SOAP body. In a communication using SOAP, a message called SOAP envelope (envelope) that includes the SOAP header and the SOAP body is described using the XML format and exchanged using a protocol such as HTTP.

The generation of a SOAP message from a command and a command response may be realized by implementing an appropriate conversion program (serializer) that is generated based on WSDL (Web Service Description Language), and serializing the data.

[0042]

The HTTP request transmission means 46 has a function for generating an HTTP request containing a transmission message generated

by the transmission message collection means 45, and transmitting the HTTP request to the HTTP server 12. It is noted that any number of transmission messages may be contained in an HTTP request, and also, transmission messages corresponding to command responses and transmission messages corresponding to client commands may be intermingled in the HTTP request on an arbitrary basis.

Therefore, the HTTP request transmission means 46 is adapted to include all the transmission messages generated by the transmission message collection means 45 in one HTTP request for transmission regardless to whether the transmission messages correspond to a command response or a client command. It is also possible to set a limit to the number of transmission messages included in one HTTP request.

[0043]

It is noted that once the transmission message collection means 45 undertakes reading of a client command or command response and so forth, the transmission of the HTTP request is performed even when there is no data to be read and consequently no transmission messages are generated. And, the transmission message collection means 45 periodically makes an attempt to read a command response or client command. For example, the readout operation may be performed every 60 minutes by a timer.

The above arrangement is made because the HTTP server 12 is unable to send information to the HTTP client 11 unless the HTTP client 11 makes a communication request to the HTTP server 12 as described in above. Even when there is no data to be sent from the HTTP client 11, the HTTP client 11 is arranged to periodically send a communication request to the HTTP server 12 to give the HTTP server 12 an opportunity to send information to the HTTP client 11 so that information necessary to be transmitted is prevented from remaining in the HTTP server 12 over a long period of time.

[0044]

It is noted that a readout operation by the transmission message

collection means 45 and the subsequent transmission of HTTP requests by the HTTP request transmission means 46 may be performed at a suitable timing aside from the periodical timing. For example, when information requiring urgent transmission is registered in any one of the pools, the client command generating means 43 or the server command execution result generating means 44 may notify this to the transmission message gathering means 45 so that the readout may be performed.

[0045]

The HTTP response receiving means 47 includes a function for receiving an HTTP response from the HTTP server 12. In the HTTP response, the receiving messages including a server command and the command ID associated with this command and the receiving messages including a response to a client command and the command ID associated with this command are intermingled on an arbitrary basis.

Herein, the receiving messages correspond to the commands or responses and the corresponding command IDs as SOAP messages.

[0046]

The receiving message distributing means 48 includes a function for allocating and registering the data contained in the HTTP response received by the HTTP response receiving means 47 into the server command pool 42 and client command pool 41.

Specifically, the server command and the command ID associated with this command are registered in a server command sheet provided in the server command pool 42. As for the response to a client command, the command ID associated with this command is collated with the command ID of each client command sheet stored in the client command pool 41 so as to specify the corresponding client command of the command response, and this command response is registered as the 'output parameter' of the corresponding client command.

The HTTP response is resolved into the respective receiving messages, and each receiving message included in this HTTP response is extracted so that data described therein is converted into a format

suitable for registration in the table. This conversion may be realized by implementing an appropriate conversion program (deserializer) that is generated based on WSDL.

[0047]

Next, Fig. 11 shows an example of an HTTP request that is to be transmitted to the HTTP server 12 by the HTTP client 11 having the above-described functions.

As is shown in Fig. 11, this HTTP request has a body portion that describes a multipart message using MIME (Multipurpose Internet Mail Extension). In each part making up the multipart message, an entity header is described and a SOAP message is embedded though detailed description in figures is omitted. In the example of Fig. 11, the HTTP body of the HTTP request is made up of independent first part, second part, third part and fourth part, divided from each other by a 'MIME_boundary'. However, the number of parts which may be included in an HTTP body is not limited to four, and any number of parts including 0 may be included.

It is also noted that the SOAP envelope embedded in the HTTP request to be transmitted includes envelopes being described with a client command or envelopes being described with a response to a server command.

[0048]

Fig. 12 shows an example of an HTTP response received by the HTTP client 11 from the HTTP server 12.

As is shown in Fig. 12, the structure of the HTTP response is identical to that of the HTTP request shown in Fig. 11 except for the HTTP header. The body portion of the HTTP response is identical to the HTTP request and describes a SOAP envelope of a multipart message in accordance with MIME. It is noted that the content of the SOAP envelope varies depending on the content of the command or command response being described.

The SOAP envelope embedded in the HTTP response to be transmitted may describe a server command or a response to a client command.

[0049]

In the following, specific examples of the parts to be described in the HTTP request or HTTP response are illustrated with reference to Figs. 13 to 16.

Fig. 13 shows an example of a part describing a client command.

In this example, the 'X-SOAP-Type' header of the entity header portion describes information indicates whether the SOAP message described in this part corresponds to a SOAP request or a SOAP response. In this case, the value 'Request' indicates that the SOAP message corresponds to a SOAP request. In other words, it is indicated that the message is a SOAP message describing a command.

Also, a 'SOAPAction' header indicates the content of a SOAP request. In this case, the content of the request is described by such URI (Uniform Resource Identifier) as 'http://www.···'. A 'SOAP header' is not attached when the SOAP message is a SOAP response'. Therefore, depending on whether or not a 'SOAPAction' header is attached, the receiving side of a message may judge whether the SOAP message is a SOAP request or a SOAP response.

[0050]

Addresses defining namespaces are indicated as attributes of the 'Envelope' tag. In this example, the namespaces specified by such URIs as 'http://www.foo.com/header' and 'http://www.foo.com/server' are defined, in addition to the namespaces defined as the norm in SOAP. Thus, an XML tag attached with a namespace prefix 'n' indicates that the tag belongs to the namespace being specified by the URI of 'http://www.foo.com/header', and an XML tag attached with a namespace prefix 'ns' indicates that the tag belongs to the namespace being specified by the URI of 'http://www.foo.com/server'.

[0051]

In the 'SOAP header', '12345' corresponding to the ID of this client command is described as the content of the 'requestID' XML tag. In the 'SOAP body', a 'trouble notification' tag is described as information indicating the method stored in the 'method name' of the

client command sheet, and the argument and other information stored in the 'input parameter' of the client command sheet are described as elements of subordinate tags such as an 'error ID' and a 'description'. In this example, the notification content of the trouble notification is described.

[0052]

Fig. 14 shows an example of a part describing a response to a client command.

In this example, the value of the 'X-SOAP-Type' header in the entity header portion is described with 'Response' to indicate that the SOAP message described in this part corresponds to a SOAP envelope describing a command response, that is a SOAP message described with a command response.

Also, the definitions of namespaces in this example are identical to those of the previous example shown in Fig. 13. Further, in the 'SOAP header', '12345' corresponding to the ID of the client command to which the response has been generated is described as the content of the 'command ID' XML tag. In the 'SOAP body', a 'trouble notification Response' tag for indicating that the part describes a response to the 'trouble notification' command is described, and the content of the command response is described in a subordinate tag. In this example, information indicating that the trouble notification has been properly received is described. Then, this information is stored in the item of 'output parameter' of the client command sheet.

[0053]

Fig. 15 shows an example of a part describing a server command.

As in the example of Fig. 13, the information 'SOAPAction' and the 'Request', the value of the 'X-SOAP-Type' header, described in the entity header portion of this part indicates that the SOAP envelope described in this part corresponds to a SOAP request. The information of the 'SOAPAction' header indicates the contents of the SOAP request.

[0054]

Addresses defining namespaces are indicated as attributes of the

'Envelope' tag, as in the case in Fig. 13. In this example, the namespaces specified by the such URIs as 'http://www.foo.com/header' and 'http://www.foo.com/client' are defined, in addition to the namespaces defined as the norm in SOAP.

In the 'SOAP header', '98765' corresponding to the ID of this server command is described as the content of the 'command ID' XML tag. Also, in the SOAP body, a 'temperature sensor value acquisition' tag is described as information indicating the method stored in the 'method name' of the server command sheet, and the argument or some other information stored in the 'input parameter' of the server command sheet is described as an element of a subordinate tag 'sensor ID'. In this example, the ID of the sensor from which the sensor value is to be acquired is described.

Further, it is noted that the server may send such a command to the client when it receives a trouble notification from the client, to determine the cause of the trouble.

[0055]

Fig. 16 shows an example of a part describing a response to a server command.

As in the example of Fig. 14, 'Response', as the value of the 'X-SOAP-Type' header, is described in the entity header portion of this part to indicate that the SOAP message described in this part corresponds to a SOAP response.

Also, as in the example of Fig. 15, '98765' corresponding to the ID of the server command to which this response is generated is described in the 'SOAP header' of this part as the content of the 'command ID' XML tag. In the SOAP body, a 'temperature sensor value acquisition Response' tag indicating that the part indicates the response to the 'temperature sensor value acquisition' command is described, and the content of the server command response is described in a subordinate tag. In this example, temperature value information provided by the sensor to which the value acquisition request has been made is described.

[0056]

In the following, processes performed by the HTTP client 11 having the above-described configurations and functions are described with reference to flowcharts shown in Figs. 17 to 21. The processes illustrated in these flowcharts may be realized by the CPU 31 of the HTTP 11, which executes the appropriate control programs.

[0057]

First of all, Fig. 17 shows a flowchart illustrating a basic operation flow of a message collection and distributing process.

The CPU 31 of the HTTP client 11 starts the process illustrated by the flowchart of Fig. 17 when it is time for the transmitting message collection means 45 to undertake readout of a client command and/or command response.

According to this process flow, first, a client command collection process is performed (S11). This corresponds to a process of collecting from the client command pool 41 client commands that are to be transmitted to the HTTP server 12, and also includes a process of generating from the collected data parts containing respective SOAP envelopes.

[0058]

Next, a server command execution result collection process is performed (S12). This corresponds to a process of collecting from the server command pool 42 command responses that are to be transmitted to the HTTP server 12, and also includes a process of generating from the gathered data parts containing respective SOAP envelopes.

Then, the parts generated in the processes of steps S11 and S12 are merged so that an HTTP request containing all these parts is generated (S13). This HTTP request is then sent to the HTTP server 12 (S14).

With regard to the processes up to this point, the CPU 31 functions as the transmitting message collection means 45 in steps S11 and S12, and as the HTTP request transmission means 46 in steps S13 and S14.

[0059]

Next, an HTTP response as a communication response to the HTTP request is received from the HTTP server 12 (S15). Then, the HTTP body of the received HTTP response is divided into parts (S16). This corresponds to a process of dividing the HTTP body into components that are separated from each other by a 'MIME_boundary', and all the parts are divided.

Then, a sequence of processes corresponding to steps S17 to S19 is successively performed for each of all the divided parts. In this process sequence, first, it is determined whether a part subjected to the process describes a server command or not (S17). Then, if it is determined that the part describes a server command, a server command registration process is performed (S18). If it is determined that the part does not describe a server command, this means that the part describes a response to a client command and, thereby, a response notification process is performed (S19).

[0060]

After performing either step S18 or S19, the process goes back to step S17, and the process sequence is repeated for a next part that is subjected to the process. Thus, the process sequence of steps S17 to S19 is performed for each of all the divided parts, and the process shown in the flowchart in FIG.17 ends when this is completed.

With regard to the processes up this point, the CPU 31 functions as the HTTP response receiving means 47 in steps S15 and S16, and as the received message distribution means 48 in steps S17 to S19.

[0061]

In the following, the operation flow of Fig. 17 is described further using flowcharts each illustrating portions of the operation flow in more detail.

FIG.18 is a flowchart illustrating a more detailed process flow of steps S11 to S14 in Fig. 17.

[0062]

According to the process, first, the CPU 31 of the HTTP client 11 collects from the client command pool 41 contents of the 'method name'

and 'input parameter' of the client command sheets indicating 'not transmitted' for the 'status' as client commands to be sent, and also collects contents of the 'command ID' as IDs of the collected client commands (S21). It is noted that when 'not transmitted' is indicated as the 'status', this means that a command generated by the client command collection means 43 has not yet been sent to the HTTP server 12. Thus, commands to be sent to the HTTP server 12 may be extracted based on the information.

[0063]

Then, processes corresponding to steps S22 to S24 are successively performed on each of all the client commands collected in the step S21. In the process, first, a client command and its corresponding command ID that are subjected to the process are converted into an XML text in which information on the client command and the command ID are included in a SOAP body and a SOAP header, respectively (S22). A SOAP envelope corresponding to a part describing the subjected client command is generated (S23). Then, the 'status' of the client command sheet describing the subjected client command is changed from 'not transmitted' to 'waiting for response' (S24). When the 'status' of the client command sheet is indicated as 'waiting for response', this means that the described command has been sent to the HTTP server 12.

[0064]

After the process sequence is completed, the CPU 31 collects the contents of the 'output parameter' of server command sheets of which the 'status' is indicated as 'process completed' as command responses to server commands that are to be transmitted to the HTTP server 12, and also collects the contents of the 'command ID' as IDs of the server commands corresponding to the collected responses (S25). When the 'status' is indicated as 'process completed', this means that the response corresponding to the server command that is generated by the server command execution result generating means 44 has not yet been sent to the HTTP server 12. Thus, the command responses to be transmitted to the HTTP server 12 may be extracted based on the

information.

[0065]

Then, processes corresponding to steps S26 to S28 are successively performed on each of all the command responses collected in the step S25. The process includes converting a subjected command response and the corresponding command ID collected therewith into an XML text that describes information on the command response and the corresponding command ID in a SOAP body and a SOAP header, respectively (S26), and generating a SOAP envelope corresponding to a part describing the subjected command response (S27). The process is identical to that of steps S22 and S23 except for the difference in the items being subjected to the processes. Next, the 'status' of the server command sheet describing the subjected command response is changed to 'response made' (S28). When 'response made' is indicated as the 'status', this means that the command response has been sent to the HTTP server 12.

[0066]

After the completion of the processes described above, the CPU 31 merges the parts generated in steps S23 and S27, generates a multipart HTTP request as shown in Fig. 11, and sends this to the HTTP server 12 (S29).

It is noted that the changing of the 'status' in steps S24 and S28 may be performed after the transmission of data is actually completed. In this way, even if communication error is generated, commands and command responses to be sent may be subjected to resending, whereby reliability of the system may be improved.

With the steps of the above, the processes pertaining to the transmission of the HTTP request are completed, and the operation moves on to steps S15 and onward of Fig. 17.

[0067]

Fig. 19 is a flowchart illustrating a more detailed operation flow of Fig. 17 for the processes of steps S15 and onward. The process step following step S29 of Fig. 18 corresponds to step S31 in this

drawing.

According to this process flow, first, the CPU 31 of the HTTP client 11 awaits the arrival of an HTTP response to the transmitted HTTP request, and then receives the HTTP response from the HTTP server 12 (S31). Upon receiving the HTTP response, the CPU 31 analyzes its HTTP body and divides it into parts (S32).

Then, a process sequence corresponding to steps S33 to S39 is successively performed on each of the divided parts.

[0068]

In this process sequence, first, it is determined whether a part subjected to the process corresponds to a server command (S33). As is described above, a server command and a response to a client command may be included in the HTTP response and, thus, a determination of whether a part corresponds to a server command or a response to a client command is made. The determination is made based on whether a SOAPAction header is described in the part, or the determination may be made based on the content of the X-SOAP-Type header.

[0069]

Then, when it is determined in step S33 that the part does not correspond to a server command, this means that the part describes a response to a client command. In this case, the XML text describing the part is analyzed and converted into data suitable for registration in the client command sheet (S34). Then, a client command corresponding to the response is searched from the client command pool 41, and the command response data is registered in the item 'output parameter' of the client command sheet of the corresponding client command (S35). It is noted that a command ID identical to the information indicated as the 'command ID' in the transmitted client command is attached to the command response and, thus, the search for the client command in step S35 may be conducted using this information as a key.

[0070]

After the data registration is completed, the 'status' of the

client command sheet is changed to 'response received' (S36). Then, the fact that a response has been received is notified to a destination registered in the 'client command execution result notifying destination' (S37). With this notification, an application program such as that for generating the client command may be informed that a response to the generated command has been received, and may perform the appropriate processes according to the response.

[0071]

For example, when an application program for generating a trouble notification generates a client command to send a trouble notification to the HTTP server 12, this command is sent to the HTTP server 12, and the HTTP server 12 may send back a command response indicating that the command has been properly received. Then, the HTTP client 11 receiving this command response may search the client command corresponding to this command response based on the command ID included in the received command response, and register this command response in association with the searched out client command. Then, the application program for generating the trouble notification that is registered as the command execution result notifying destination of this command is notified of the fact that a response to this command has been received. The application program for receiving this notification may refer to the client command sheet and acquire the execution result of the generated command from the item 'output parameter' of this client command sheet.

After the processes up to step S37 are completed for the subjected part, the same processes starting from step S33 are repeated for a next part if such part exists.

[0072]

On the other hand, when it is determined in step S33 that the subjected part corresponds to a server command, the XML text describing the part is analyzed and converted into data suitable for registration in the server command sheet (S38). Then a server command sheet for this server command is created and this server command sheet containing

the server command and its corresponding command ID is registered in the server command pool 42 (S39). The content of the server command is registered in the items 'method name' and 'input parameter' of the server command sheet, and the command ID described in the part is registered in the item 'command ID' of the server command sheet. In the item 'server command notifying destination', reference information to the application program with which the method registered in the 'method name' is executed, for example, is registered, this reference information being determined based on correspondence information between a method and an application program, for example, that is provided beforehand. It is noted that the initial value of the 'status' corresponds to 'not processed', and the initial value of the 'output parameter' corresponds to 'NULL'.

[0073]

After the processes up to step S39 are completed, the processes starting from step S33 are repeated for a next subjected part if such part exists.

The process illustrated in the flowchart of Fig. 19 ends when the processes from step S33 to step S39 are performed on each of all the parts.

By performing the above-described processes, the HTTP client 11 is able to send to the HTTP server 12 an operation request to be transmitted to the HTTP server 12 together with an operation response to an operation request received from the HTTP server 12 in one bundle. Also, the HTTP client 11 is able to receive from the HTTP server 12 an operation request sent by the HTTP server 12 together with an operation response to the operation request that has been sent to the HTTP server 12 in one bundle.

[0074]

It is noted that in the present embodiment, the parts to be transmitted are generated and merged together before they are transmitted, and the parts are received as a whole after which they are divided into parts for further processing; however, the present

invention is not limited to this arrangement.

As for the transmission, the HTTP header may first be transmitted, after which parts are successively transmitted each time they are generated, and notification data may be sent after the transmission of the parts is completed. Even in such arrangement, as long as the data being transmitted during the process is one logically successive HTTP request having one HTTP header, the transfer of data may be transferred in one session with one negotiation process. Thus, an effect similar to that from merging the parts together before transmitting the HTTP request may be obtained as well. Further, since the required memory capacity of the buffer for storing data to be transmitted may be reduced, a low cost communication apparatus may be arranged to handle a large quantity of data.

Also, at the receiving side, the processes for each part may be successively performed each time a part is received. As in the transmission, the required memory capacity may be reduced.

[0075]

In the following, processes pertaining to an execution of a server command are described.

Fig. 20 is a flowchart illustrating an example of such processes.

The process steps of Fig. 20 pertaining to an execution of a server command are performed after step S39 of Fig. 19. That is, these steps may be performed after the server command is registered in the server command pool 42. Herein, the CPU 31 of the HTTP client 11 functions as the server command execution result generating means 44.

[0076]

In the process, first, an application program and so forth is called based on the information corresponding to the 'server command notifying destination' of the server command sheet describing the registered server command, and data corresponding to the 'method name' and 'input parameter' are transferred to the application program so that a process pertaining to the server command is performed (S41). Although it is not shown in this flowchart, the process pertaining

to the server command is executed by the CPU 31 separately.

After this process step is completed, the execution result is registered in the item 'output parameter' of the server command sheet (S42). At the same time, the 'status' of the server command sheet is changed to 'process complete' which indicates that the process is completed (S43). Then, the operation goes back to the process flow of Fig. 19.

By means of the process steps as in above, the server command may be executed and appropriate measures may be taken so that the execution result of the server command may be in a state ready for transmission to the HTTP server 12 as the command response.

[0077]

A flowchart shown in Fig. 21 illustrates another exemplary process flow pertaining to the execution of the server command that is performed independently from the process flow of Fig. 19. Herein, the CPU 31 of the HTTP client 11 also functions as the server command execution result generating means 44.

In performing the process steps of Fig. 21, the CPU 31 initiates the process shown in the flowchart of Fig. 21 at suitable timings.

[0078]

In the process of Fig. 21, it is determined whether a server command sheet of which the 'status' is indicated as 'waiting for process' exists or not (SX1), and if it does not exist, the process is in a waiting status until such a server command sheet is added. When such server command sheets are found, the 'status' of the server command sheet is changed to 'in process' as one of the command sheets is subjected for processing (SX2).

After the process in above, the processes corresponding to the steps S41 to S43 as similar to the processes in Fig. 20 are performed so that the server command subjected for the process and described in the server command sheet is executed. After the completion thereof, the process is repeated returning to the step SX1.

[0079]

It is noted that the above processes may be performed simultaneously using a plurality of threads (e.g., four threads). Herein, since the 'status' of a server command sheet that is once subjected to the process is no longer indicated as 'waiting for process', the server command sheet is prevented from being subjected to a process more than once even if a process is started simultaneously in a plurality of threads.

[0080]

By performing the above-described processes, each server command may be executed at an arbitrary timing and, thereby, even when the execution of one command takes a long time, its subsequent processes may not necessarily be affected by this delay. Further, the execution results are successively adjusted for transmission to the HTTP server 12 as command responses according to the order in which the execution of each command is completed.

In the above, the description of the processes pertaining to the communication method of the present invention to be performed in the HTTP client 11 has been completed.

[0081]

Herein, though drawing and description have been omitted, in the HTTP server 12 too, the part related to the communication method of the present invention has a similar hardware configuration and a function configuration with that of the HTTP client 11 respectively, and a similar process is performed. However, since there is a difference in function as a server and a client, the following point is different with that for the HTTP client 11.

[0082]

First, the HTTP server 12 includes an HTTP response transmission means and an HTTP request receiving means instead of the HTTP request transmission means 46 and the HTTP response receiving means 47. Also, a client command is received and a response to the command is transmitted, and a server command is transmitted and a response to the server command is received. Then, the timing in which the

transmission message collection means 45 tries reading of a command and a command response is when an HTTP request has been received and then analysis of the HTTP request has been completed. This is because the HTTP server 12 transmits a command and a command response to the HTTP client 11 as a communication response to the HTTP request.

The communication method of the present invention is realized by performing such process by the HTTP server 12.

[0083]

Then, according to the communication method of the present invention which has been described in above, since an operation request to be transmitted from a transmitting side to the communication counterpart and an operation response to the operation request received from the communication counterpart may be transmitted to the communication counterpart in one batch, establishing a communication connection may not be necessary by performing negotiation separately concerning to transmission to each an operation request and an operation response, whereby communication overhead may be reduced and communication efficiency may be improved.

[0084]

Also, an operation request and an operation response are changed to serialized data respectively, and changed to a transmission message described in a structure language format, wherein a differently formatted operation request and an operation response may easily connected and may be transmitted logically as one transmission content.

Also, for a receiving side, a received content may be divided into each individual message easily and an approximate process may be performed depending on whether the message is an operation request or an operation response.

[0085]

Further, by arranging that communication is performed by issuing a communication request from always an apparatus side and transmission of an operation request from the communication counterpart to a communication request side is performed as a response to the

communication request, transmission and receiving of the operation request and operation response may be performed without concerning a firewall, even if this is a communication system in which an apparatus for issuing an communication request is arranged inside of the firewall. Also, since a communication request and a communication response is corresponding each other, timing of a communication level may be easily managed.

Also, in this case, if communication request is arranged to be performed from one side of the above apparatus periodically to the communication counterpart, a situation in which transmission of information from the inside to outside of the firewall is stagnated for a long time may be prevented.

[0086]

[Embodiment example applying SMTP: Figs. 22 to 27]

In the following, a reference example in which SMTP is used as the communication protocol is described. It is noted that many common points exist between this reference example and the previously-described embodiment using HTTP as the communication protocol. Thus, parts of the description of this reference example that are identical to those of the embodiment using HTTP are omitted or simplified, and an emphasis is put on features of this reference example that are different from those of the embodiment using HTTP.

Fig. 22 is a block diagram showing an exemplary configuration of a communication system implementing the above embodiment example using SMTP.

[0087]

As shown in Fig. 22, the communication system includes a LAN_A connecting a communication apparatus A and a mail server A', a LAN_B connecting a communication apparatus B and a mail server B', and the Internet 13 connecting the LAN_A and LAN_B via a firewall A of the LAN_A and a firewall B of the LAN_B. A mail server A and a mail server B are provided in the LAN_A and LAN_B, respectively, at positions enabling access from outside via the respective firewalls. It is

noted that the communication apparatus A corresponds to a first communication apparatus, and the communication apparatus B corresponds to a second communication apparatus.

[0088]

In a communication using SMTP, the transfer of information between the communication apparatus A and the communication apparatus B is realized by electronic mail. Specifically, for example, when information is to be sent from the communication apparatus B to the communication apparatus A, the communication apparatus B outputs an electronic mail addressed to the communication apparatus A, which is first sent to the mail server B', as is indicated by the dashed line arrows in Fig. 22. Then, the electronic mail is transferred via the mail servers B', B, and A, respectively, before being transferred to the mail server A' to which the communication apparatus A has direct access. Although illustration in a drawing is omitted, transfer is performed via another mail server between the mail server B and the mail server A.

Then, as indicated by the dash-dotted line arrow, the communication apparatus A may periodically access the mail server A' to receive electronic mail addressed to thereto. In this way, information transmission from the communication apparatus B to the communication apparatus A is completed. On the other hand, in transmitting information from the communication apparatus A to the communication apparatus B, a reverse procedure of the above-described procedure may be performed. In other words, the communication apparatus A and the communication apparatus B operate in a similar manner with respect to information transmission. It is noted that the mail server A' and mail server B' do not necessarily have to be arranged, and the communication apparatus A may communicate directly with the mail server A and the communication apparatus B may communicate directly with the mail server B. Also, a mail server for receiving mails may be different from a mail server for mail transmission.

[0089]

In such a system, each LAN implements a mail server at a position that allows access from outside so that electronic mail may be sent thereto via the firewall.

In the reference example, the communication apparatus A and the communication apparatus B are able to transmit information to/from each other even though they do not communicate through direct negotiation.

[0090]

As with the first and second communication apparatuses of Fig. 1, the communication apparatus A and communication apparatus B implement application programs for controlling and managing each other. Using RPC, each communication apparatus is arranged to send an 'operation request' calling for the execution of the process for a method of an application program implemented at the other side, and acquire an 'operation response' corresponding to the execution result of the requested process.

[0091]

Fig. 23 shows relations between the operation request and the operation response. As is shown, in the operation request level, the communication apparatus A and the communication apparatus B operate in a similar manner as with the HTTP client 11 and the HTTP server 12. However, in the case of using SMTP, the communication apparatus A and the communication apparatus B operate in a similar manner even in the communication level, this being different from the relation of the HTTPs.

[0092]

Fig. 24 shows an exemplary communication sequence in this communication system. The communication sequence shows an embodiment of the communication method of the present invention.

As described earlier, in this communication system, communication between the communication apparatus A and the communication apparatus B is performed using electronic mail. This electronic mail includes information on a sender and a destination, and a reply may be sent

back to the sender from the destination. However, in this case, the first mail and the reply mail are independent from each other; that is, in this example there is no relation similar to the communication request-communication response relation found in the embodiment using HTTP.

Thus, either of the two communication apparatuses is able to initiate the communication, and the electronic mail does not necessarily have to be sent back and forth. In the following example, a case will be described in which a total of three electronic mails are transmitted/received back and forth starting with the communication apparatus A sending an electronic mail to the communication apparatus B.

[0093]

In these electronic mails, an operation request (command) to a destination (address) and an operation response (command response or simply 'response') to the command received from the destination are described. This case applies whether the sender corresponds to the communication apparatus A or the communication apparatus B.

Thus, for example, communication apparatus A command a may be described in electronic mail x to be transmitted to the communication apparatus B, and subsequently, the corresponding command response from the communication apparatus B may be described in electronic mail y to be sent to the communication apparatus A. Also, communication apparatus B command c may be described in the electronic mail y and, subsequently, the corresponding command response from the communication apparatus A may be described in electronic mail z to be sent to the communication apparatus B.

[0094]

It is noted that any number (may be 0) of operation requests and operation responses to the operation request may be described in one electronic mail. Further, the content described in one electronic mail corresponds to one message and is logically sent as one batch. In this way, the number of electronic mails for transmitting

information can be reduced so that the communication overhead is reduced and communication efficiency is improved.

[0095]

Next, in the following, functional configurations and process steps for realizing the processes of merging the commands and command responses and separating these parts in the communication apparatus A and communication apparatus B will be described. It is noted that the hardware configurations of these apparatuses may be identical to the configuration illustrated in Fig. 7 for the HTTP client 11 and HTTP server 12.

[0096]

Fig. 25 is a functional block diagram illustrating a functional configuration of the communication apparatus A for performing the processes pertaining to the commands and command responses corresponding to Fig. 8.

As is shown, the communication apparatus A includes a communication apparatus A command pool 51, a communication apparatus B command pool 52, communication apparatus A command generating means 53, and communication apparatus B command execution result generating means 54, which correspond to the client command pool 41, the server command pool 42, the client command generating means 43, and the server command execution result generating means 44, respectively, as shown in Fig. 8. The differences in the names of these corresponding functions are due to the differences in the names of the apparatuses.

[0097]

The transmission message collection means 45 and the receiving message distribution means 48 correspond to the means with the same names as shown in Fig. 8. It is noted that in this example, a data format corresponding to SMTP is different from the case shown in Fig. 8. However, the individual SOAP messages corresponding to the commands and command responses described are identical to those used in the embodiment of Fig. 8.

The mail transmitting means 56 and the mail receiving means 57

correspond to transmitting means and receiving means, but are different from the HTTP request transmitting means 46 and the HTTP response receiving means 47 since the protocol being used for transmission and reception of data in the two examples is different.

[0098]

The mail transmitting means 56 has the function of generating an electronic mail addressed to the communication apparatus B that includes the outgoing message generated by the outgoing message gathering means 45, and sending this to the mail server A'. After this, the communication apparatus A does not interfere with the transmission of this electronic mail. As in the embodiment of Fig. 8, any number of outgoing messages may be included in one electronic mail, and the outgoing message may correspond to a command response or a communication apparatus A command. The outgoing messages corresponding to a command response and outgoing messages corresponding to a communication apparatus A command may be intermingled on an arbitrary basis.

[0099]

The mail receiving means 57 has the function of checking to see whether there are any newly arrived mails, and receiving the newly arrived mails if they exist. The electronic mail to be received may include any number of receiving messages describing a communication apparatus B command and its associated command ID, and receiving messages describing a response to a communication apparatus A command and its associated command ID, that are intermingled with each other on an arbitrary basis.

[0100]

Fig. 26 shows an exemplary electronic mail to be sent to the communication apparatus B from the communication apparatus A having the above-described functions.

This electronic mail, like the HTTP request of Fig. 11, has a body portion that describes a multipart message using MIME (Multipurpose Internet Mail Extension). In each part making up the multipart

message a SOAP envelope is embedded. Thus, the body portion of this electronic mail is identical to the HTTP body of the HTTP request.

[0101]

However, the header portion of this electronic mail is different from the HTTP header in that information corresponding to items 'From' for indicating the sender address of the electronic mail, 'To' for indicating the destination address, and 'Subject' for indicating the title, for example, are described therein.

As for the electronic mail to be sent to the communication apparatus A from the communication apparatus B, the information content described in the items 'From' and 'To' are exchanged.

Also, the contents of the SOAP envelopes described in the electronic mails are identical to those described in the HTTP request or HTTP response. However, in the entity header portion of each part, different information may be described since the data encoding method used in this example is different from that used in the embodiment using HTTP.

[0102]

In the following, processes executed in the communication apparatus A having the above-described configurations and functions will be described with reference to flowcharts shown in Figs. 27 and 28. It is noted that the processes shown in these flowcharts are executed by a CPU that is implemented in the communication apparatus A and executes suitable control programs.

[0103]

Fig. 27 is a flowchart illustrating a basic operation flow of processes executed upon message transmission.

First, the CPU of the communication apparatus A starts the process flow of Fig. 27 at an appropriate timing for the outgoing message gathering means 45 to undertake a readout of communication apparatus A commands and/or command responses.

Then, a communication apparatus A command gathering process is performed (S51). This corresponds to a process of gathering from the

communication apparatus A command pool 51 communication apparatus A commands to be sent to the communication apparatus B, and also includes a process of generating SOAP envelope parts based on the gathered data.

[0104]

Then, a communication apparatus B command execution result gathering process is performed (S52). This corresponds to a process of gathering from the communication apparatus B command pool 52 command responses to be sent to the communication apparatus B, and also included a process of generating SOAP envelope parts based on the gathered data.

Then, the parts generated in the processes of steps S51 and S52 are merged so that an electronic mail containing these parts is generated (S53). This electronic mail is then addressed to the communication apparatus B and sent thereto (S54) and, hereby, the message transmission process is completed.

[0105]

In executing the above-described processes, the CPU31 functions as the transmission message collection means 45 in steps S51 and S52, and as the mail transmitting means 56 in steps S53 and S54.

It is noted that these processes correspond to steps S11 to S14 shown in Fig. 17. However, in the case of using SMTP, the transmission of electronic mail and the reception of electronic mail each correspond to individual processes and a relation such as that between the communication request and the communication response in the data transmission/reception between the HTTP client 11 and HTTP server 11 does not exist in this example. Thus, the transmission process ends here, and another separate process shown in Fig. 28 is performed in receiving the electronic mail.

[0106]

Fig. 28 is a flowchart illustrating a basic operation flow of a message receiving process.

The CPU of the communication apparatus A periodically accesses the mail server A' and starts the process illustrated in the flowchart

of Fig. 28 when it detects a newly arrived electronic mail addressed to the communication apparatus A.

In this process, first, the CPU receives the newly arrived electronic mail (S61). Then, the body (text) of the received electronic mail is divided into parts (S62). This corresponds to a process of dividing the body into components that are separated from each other by a 'MIME_boundary'.

[0107]

Then, a sequence of processes corresponding to steps S63 to S65 is successively performed for each of the divided parts. In this process sequence, first, it is determined whether a part subjected to the process describes a communication apparatus B command (S63). Then, if it is determined that the part describes a communication apparatus B command, a communication apparatus B command registration process is performed (S64). On the other hand, if it is determined that the part does not describe a communication apparatus B command, this means that the part describes a response to a communication apparatus A command and, thereby, a response notification process is performed (S65).

[0108]

After performing either step S64 or S65, the process goes back to step S63, and the process sequence is repeated for a next part that is subjected to the process. Thus, the process sequence of steps S63 to S65 is performed for each of the divided parts, after which the message receiving process of Fig. 28 ends.

In executing the above-described processes, the CPU31 functions as the mail receiving means 57 in step S61, and as the received message distribution means 48 in steps S62 to S65.

These processes correspond to steps S15 to S19 of Fig. 17.

[0109]

It is also noted that the processes relating to the execution of a communication apparatus B command in the communication apparatus A are identical to the processes relating to the execution of the server

command described with reference to Figs. 20 and 21.

In the above, the processes relating to the transmission of a command and a command response performed in the communication apparatus A have been described.

As for the functions and processes of the communication apparatus B, the concepts of the communication apparatus A command and the communication apparatus B command may be switched in the above descriptions of the communication apparatus A.

[0100]

As described above, the communication method of the present invention may be applied to communications performed by using a protocol in which transmission of information does not necessarily correspond to receiving information as a response to this transmission. Even in such case, an operation request to be sent from a sender to a communication counterpart, and an operation response to an operation request that has been received from the communication counterpart may be collectively sent to the communication counterpart in one batch so that the operation request and the operation response do not have to be transmitted in separate units. In this way, the communication overhead may be reduced and communication efficiency may be improved.

Also, by using electronic mail for data transmission and receiving, information may be easily transferred within the firewalls.

[0164]

[Modification Example of Each Embodiment Example: Fig. 29]

In the following, an exemplary modification of the above-described each embodiment example is described.

In the above-described each embodiment example, an example of the present invention applied to a communication system that is made up of two communication apparatuses; however, this has been done to simplify the description and the present invention is not limited to these examples. For example, the present invention may be applicable to a communication system made up of a greater number of communication apparatuses.

For example, another exemplary communication system that uses HTTP may be applied to a communication system shown in Fig. 29.

[0112]

The communication system configures a remote management system for managing a target managing apparatus 100 by means of a managing apparatus 102 by establishing a connection between the managing apparatus 102 having communication functions and the target managing apparatus 100 having also communication functions via a network including the Internet 13.

The target managing apparatus 100 may correspond to various types of electronic apparatuses having communication functions including, for example, a printer, a scanner, a copier, a digital multi functional imaging apparatus implementing more than one of the above functions, a network appliance, a vending machine, medical equipment, a power source apparatus, an air conditioning system, measuring systems for gas, water, or electricity, a computer that can be connected to a network.

[0113]

The remote management system includes an intermediate device 101 which corresponds to a communication apparatus for mediating the communication between the managing apparatus 102 and the target managing apparatus 100. Thus, the managing apparatus 102 and the target managing apparatus 100 communicate with each other via the intermediate device 101.

The intermediate device 101 and the target managing apparatus 100 may be arranged in various structures in configurations thereof according to the environment in which those apparatuses are used. For example, in environment A shown in Fig. 29, the target managing apparatuses 100a and 100b are arranged under the control of the intermediate device 101a which is capable of establishing a connection with the managing apparatus 102 using HTTP. On the other hand, in environment B, four units of the target apparatus 100 are provided and, thereby, the load may be too large for one intermediate device

101 to subordinate the managed apparatuses.

[0114]

For this reason, the intermediate device 101b which may establish a connection with the managing apparatus 102 1 using HTTP subordinates not only the target managing apparatuses 100c and 100d, but also the other intermediate device 101c. Further, the intermediate device 101c subordinates the target apparatuses 100e and 100f. Information generated from the managing apparatus 102 to conduct remote management on the managed apparatuses 100e and 100f passes through the intermediate device 101b, and then the intermediate device 101c, which corresponds to a subordinate node of the intermediate device 101b, to reach the target apparatus 100e or 100f. Also, for security reasons, firewall 14 is implemented in each environment.

[0115]

By handling the intermediate device 101 as the HTTP client and the managing apparatus 102 as the HTTP server, the communication method of the present invention may be applied to the remote management system.

In the case, when the managing apparatus 102 is to send a command to the target apparatus 100, the managing apparatus 102 may first send a command 'send command to the target managing apparatus 100', and the intermediate device 101 may be arranged to send the command to the target managing apparatus 100 as an operation corresponding to the command received from the managing apparatus 102. Alternatively, the managing apparatus 102 may designate any one of the target managing apparatuses 100 as a destination of a command and then send this command to the intermediate device 101, and the intermediate device 101 may be arranged to send a command received from the managing apparatus 102 that is addressed to a destination other than itself to the designated destination.

[0116]

When transmission/reception of commands and command responses is performed among three or more nodes, information on the sender and

destination of the command may be included in a message corresponding to the command or command response so that the sender and destination of the command may be determined, and preferably, this information is described and managed in a command sheet as well.

[0117]

By implementing a mail server in this system, the reference example using SMTP may also be applied to this system.

The present invention is not limited to these embodiments, and further variations and modifications may be made without departing from the scope of the present invention. For example, the client command sheet and server command sheet respectively registered in the client command pool 41 and the server command pool 42 may be described as XML formatted documents.

A limit may be imposed on the amount of information contained in a command and a command response to be transmitted to the HTTP server 12. Especially, by limiting the amount of receiving information of the command, usage amount of memory may be contained when the receiving side is an apparatus having limited memory capacity.

[0118]

In the above described embodiment example, SOAP is used as the upper protocol for realizing RPC and RPC is realized with an application directly operating the pool. However, other protocols such as CORBA (Common Object Request Broker Architecture) or JAVA (Registered Trademark) RMI (Remote Method Invocation) may be used as well between the application and the pool.

In other words, according to the above embodiment example, the exchange of a command and a response to the command between the HTTP client 11 and the HTTP server 12 is realized by a SOAP message described in XML; however, the present invention is not limited to this arrangement and data may be described in other formats as well.

[0119]

In the present embodiment, unique protocols are used in addition to protocols according to the SOAP standard so that the SOAP envelopes

included in the HTTP request or HTTP response may be handled as independent parts. Using the SOAP attachment covered by the SOAP standard protocol, the SOAP envelope corresponding to the first part of the HTTP response may be arranged to contain links to subsequent SOAP envelopes corresponding to the second part and onward so that they are associated when handed down. This arrangement may similarly be applied to the reference example using electronic mail and SMTP.

[0120]

Further, the present embodiment describes in a case that HTTP is used for data transmission as the subordinate protocol with respect to the upper protocol such as SOAP. However, other protocols such as FTP (File Transfer Protocol) may also be used as the subordinate protocol.

The configuration of the communication system according to the present invention is not limited to the examples described in above.

[Utility in the Industry]

[0121]

As described in above, according to the communication method of the present invention, in a case in which a plurality of communication apparatuses transmit/receive operation requests and operation responses to the received operation requests for each other, communication efficiency may be improved.

Therefore, a communication system bearing a small communication load may be configured by applying the invention to such communication system in which a plurality of communication apparatuses receive/transmit operation requests and operation responses to the received operation requests.

[Brief Description of the Drawings]

[0122]

[Fig. 1]

Fig. 1 is a diagram illustrating an exemplary configuration of a communication system using the communication method of the present invention.

[Fig. 2]

Fig. 2 is a diagram illustrating a relationship of an operation request and an operation response according to the communication system shown in Fig. 1.

[Fig. 3]

Fig. 3 is a diagram illustrating an exemplary configuration of a communication system using an embodiment of the communication method of the present invention in a case when HTTP is used as a communication protocol.

[Fig. 4]

Fig. 4 is a diagram illustrating a relationship of an operation request and an operation response in the communication system shown in Fig. 3.

[Fig. 5]

Fig. 5 is a diagram illustrating an example of a communication sequence in the communication system shown in Fig. 3.

[Fig. 6]

Fig. 6 is a diagram illustrating another example of the above communication sequence.

[Fig. 7]

Fig. 7 is a diagram illustrating an exemplary configuration of hardware of the HTTP client and HTTP server shown in Fig. 3.

[Fig. 8]

Fig. 8 is a functional block diagram illustrating functional configuration for performing a process related to a command and a command response in functions of the HTTP client shown in Fig. 3.

[Fig. 9]

Fig. 9 is a diagram illustrating an example of data configuration in a client command sheet to be stored in the client command pool shown in Fig. 8.

[Fig. 10]

Fig. 10 is a diagram illustrating an example of data configuration in a server command sheet to be stored in the server command pool shown

in Fig. 8.

[0123]

[Fig. 11]

Fig. 11 is a diagram illustrating an example of a HTTP request which the HTTP client shown in Fig. 3 transmits to the HTTP server.

[Fig. 12]

Fig. 12 is a diagram illustrating an example of a HTTP response which the HTTP client shown in Fig. 3 receives from the HTTP server.

[Fig. 13]

Fig. 13 is a diagram illustrating an example of a part in which a client command is described.

[Fig. 14]

Fig. 14 is a diagram illustrating an example of a part in which a response to a client command is described.

[Fig. 15]

Fig. 15 is a diagram illustrating an example of a part in which a server command is described.

[Fig. 16]

Fig. 16 is a diagram illustrating an example of a part in which a response to a server command is described.

[Fig. 17]

Fig. 17 is a flowchart illustrating a functional operation of a message collection and distribution process in the HTTP client shown in Fig. 3.

[Fig. 18]

Fig. 18 is a flowchart illustrating in more detail a process of parts in the steps S11 to S14 of Fig. 17.

[Fig. 19]

Fig. 19 is a flowchart illustrating in more detail a process of parts in the steps on and after S15 of Fig. 17.

[Fig. 20]

Fig. 20 is a flowchart illustrating an example of a process related to execution of a server command in the HTTP client shown in Fig. 3.

[0124]

[Fig. 21]

Fig. 21 is a flowchart illustrating another example of the above process.

[Fig. 22]

Fig. 22 is a diagram illustrating a configuration of a communication system using an embodiment of the communication method of the present invention in a case in which SMTP is applied as a communication protocol.

[Fig. 23]

Fig. 23 is a diagram illustrating a relation between an operation request and an operation response in the communication system shown in Fig. 22.

[Fig. 24]

Fig. 24 is a diagram illustrating an example of a communication sequence in the communication system shown in Fig. 22.

[Fig. 25]

Fig. 25 is a functional block diagram corresponding to Fig. 8, illustrating a configuration of a function for performing a process related to a command and a command response in the functions of the communication apparatus A shown in Fig. 22.

[Fig. 26]

Fig. 26 is a diagram illustrating an example of an electronic mail which the communication apparatus A sends to the communication apparatus B shown in Fig. 22.

[Fig. 27]

Fig.27 is a flowchart illustrating a fundamental operation of a process upon transmitting a message in the communication apparatus A shown in Fig. 22.

[Fig. 28]

Fig. 28 is a flowchart illustrating a flowchart of a fundamental operation of a process upon transmitting a message as well.

[Fig. 29]

Fig. 29 is a diagram illustrating an example of another configuration of the communication system using the communication method of the present invention.

[Description of Numerals]

[0125]

1 :First communication apparatus 2 :Second communication apparatus

10 :Network 11 :HTTP client

12 :HTTP server 13 :Internet

14 :Firewall 31 :CPU

32 :ROM 33 :RAM

34 :SD card 35 :NIC

41 :Client command pool

42 :Server command pool

43 :Client command generation means

44 :Server command execution result generation means

45 :Transmission message collection means

46 :HTTP request transmission means

47 :HTTP response receiving means

48 :Receiving message distribution means

51 :Communication apparatus A command pool

52 :Communication apparatus B command pool

53 :Communication apparatus A command generation means

54 :Communication apparatus B command execution result generation means

56 :Mail transmission means 57 :Mail receiving means

100 :Target managing apparatus 101 :Intermediate device

102 :Managing apparatus

[Name of Document] Drawings

[Name of Document] Abstract of the Disclosure

[Abstract]

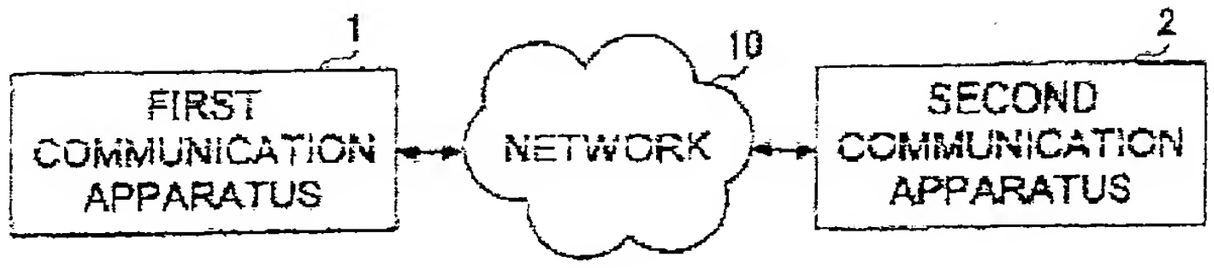
[Objectives of the Invention] To improve communication efficiency when a plurality of communication apparatuses transmit/receive operation requests and operation responses to the received operation requests for each other.

[Means for Achieving the Objectives]

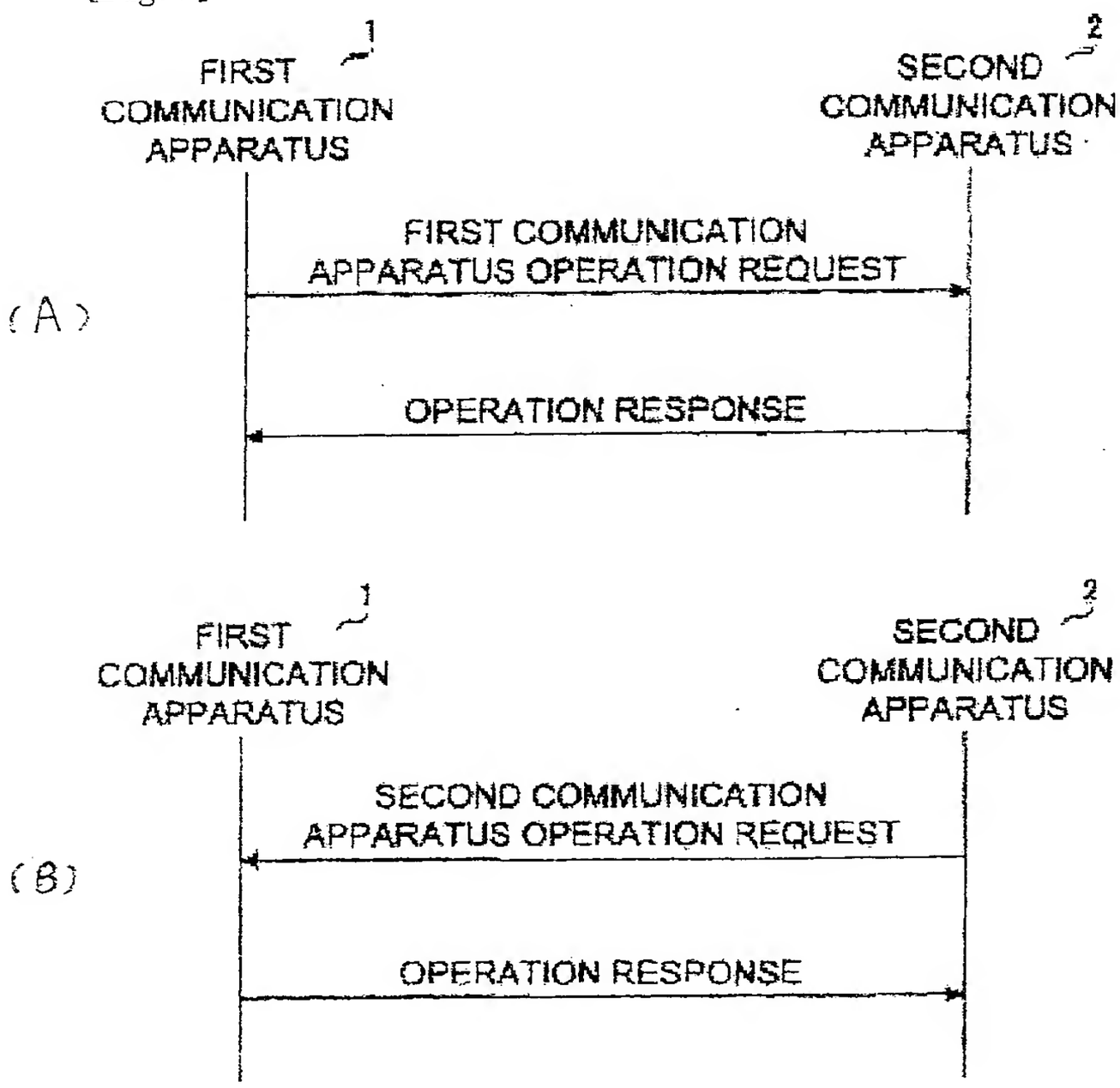
An HTTP client 11 transmits to an HTTP server 12, a client command corresponding to be transmitted to the HTTP server 12 and a response to a server command having received from the HTTP server 12 in one batch as one HTTP request, and the HTTP client 12 transmits to the HTTP client 11, a server command to be transmitted to an HTTP client and an operation response to the client command received from the HTTP client in one batch as one HTTP response. In the case, each command may be a function call, and a response to the command may be a result of the execution of the function called by the function call.

[Selected Drawing] Fig. 5

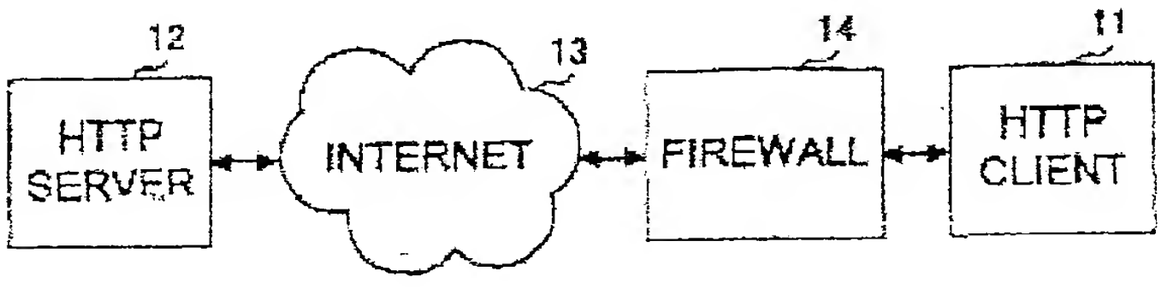
[Fig. 1]



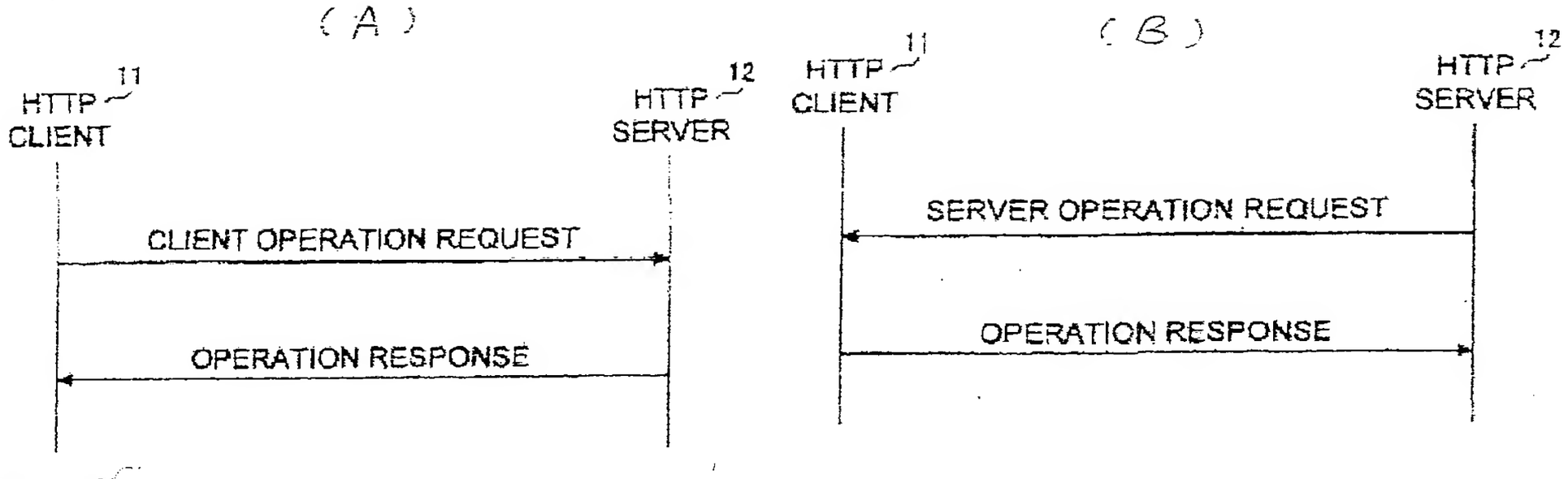
[Fig. 2]



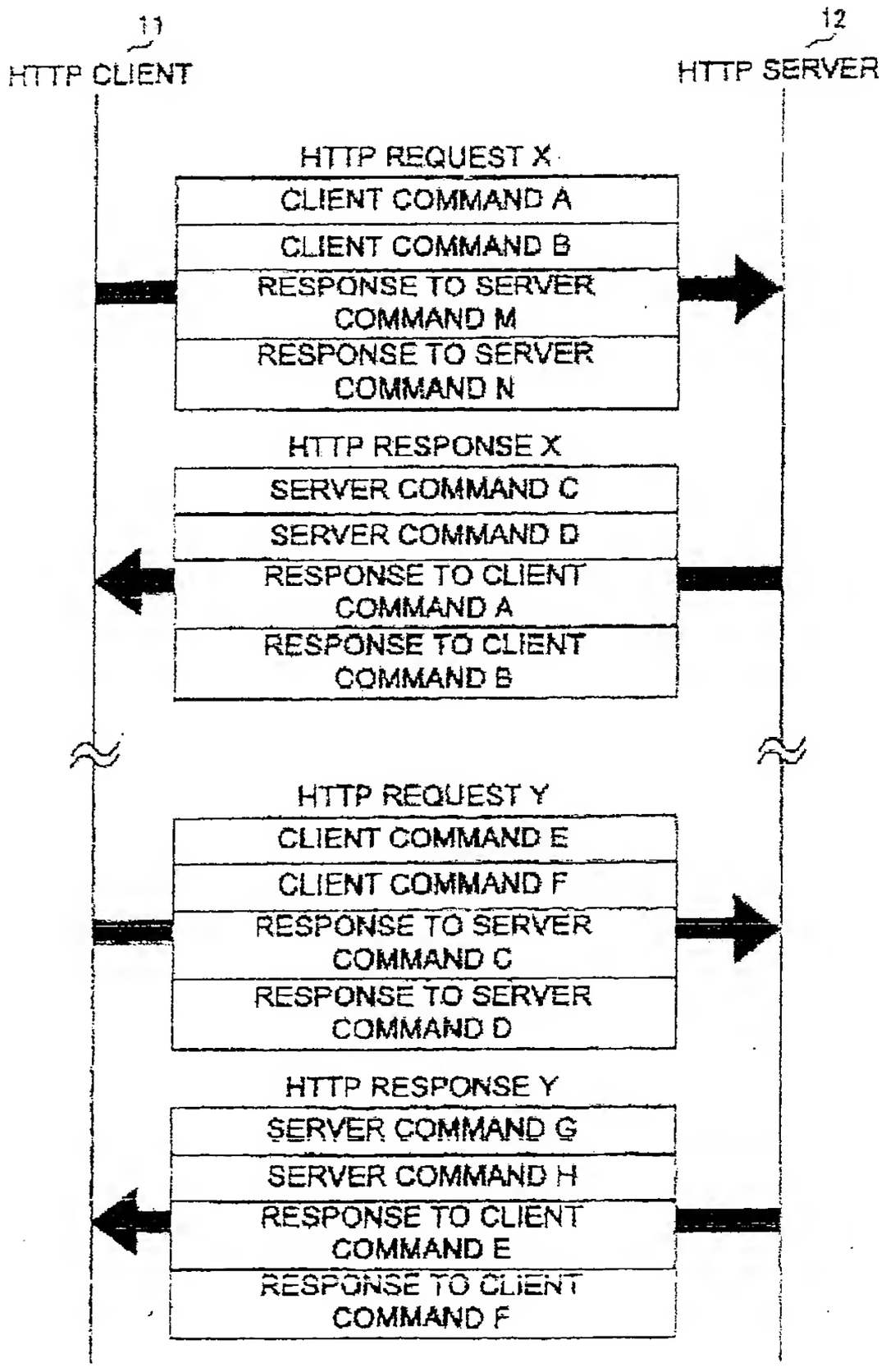
[Fig. 3]



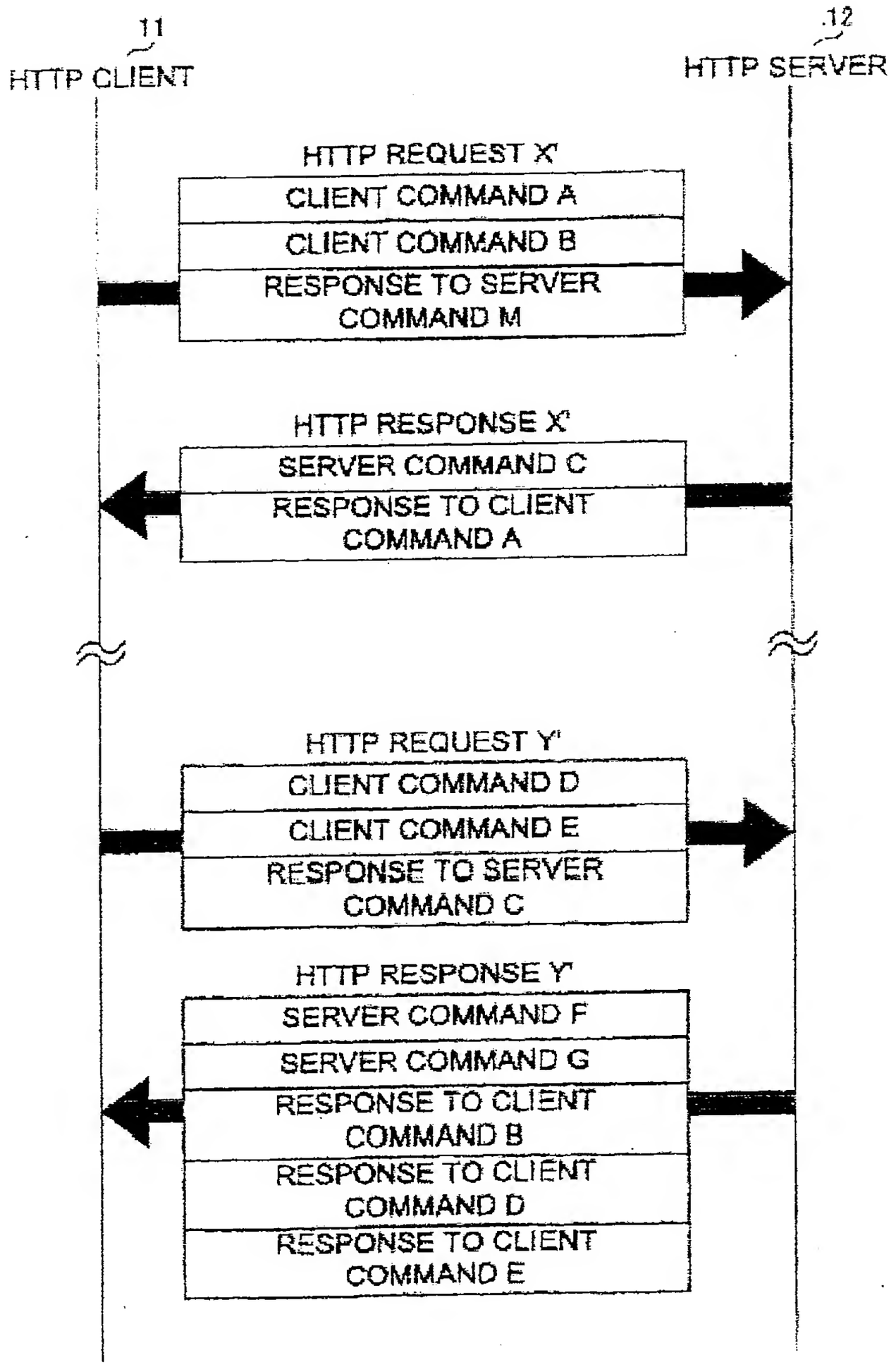
[Fig. 4]


(A)
HTTP CLIENT
11
HTTP SERVER
12
CLIENT OPERATION REQUEST
OPERATION RESPONSE
(B)
HTTP CLIENT
11
HTTP SERVER
12
SERVER OPERATION REQUEST
OPERATION RESPONSE

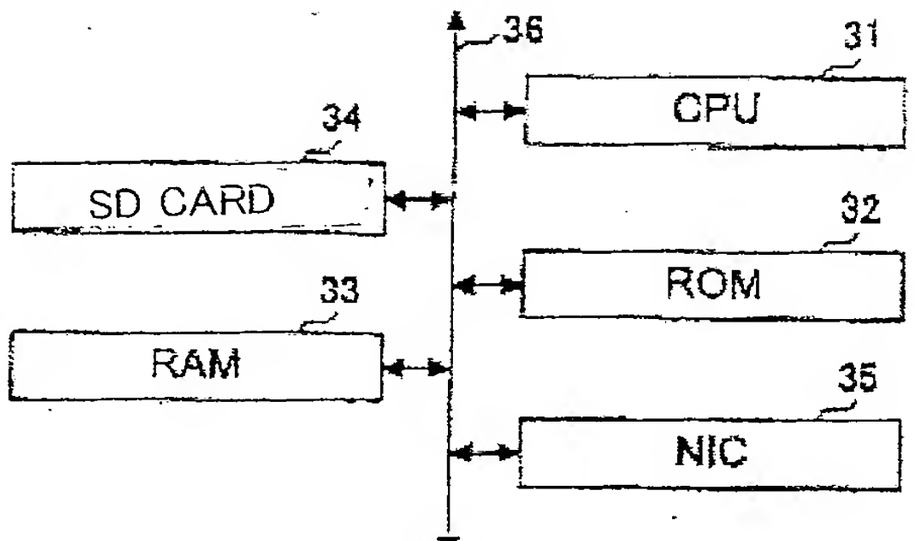
[Fig. 5]


HTTP CLIENT
11
HTTP SERVER
12
HTTP REQUEST X
CLIENT COMMAND A
CLIENT COMMAND B
RESPONSE TO SERVER COMMAND M
RESPONSE TO SERVER COMMAND N
HTTP RESPONSE X
SERVER COMMAND C
SERVER COMMAND D
RESPONSE TO CLIENT COMMAND A
RESPONSE TO CLIENT COMMAND B
HTTP REQUEST Y
CLIENT COMMAND E
CLIENT COMMAND F
RESPONSE TO SERVER COMMAND C
RESPONSE TO SERVER COMMAND D
HTTP RESPONSE Y
SERVER COMMAND G
SERVER COMMAND H
RESPONSE TO CLIENT COMMAND E
RESPONSE TO CLIENT COMMAND F

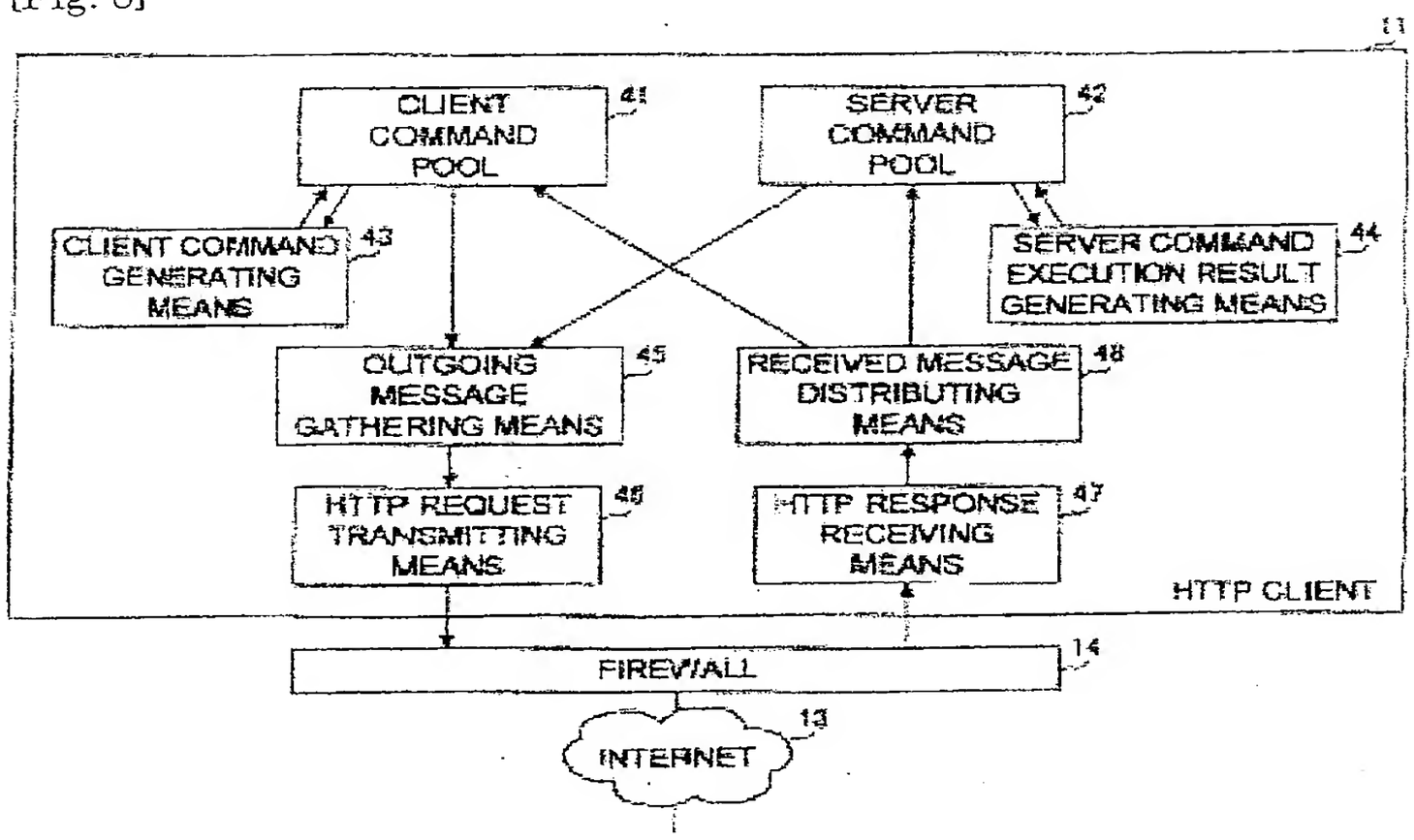
[Fig. 6]


11
HTTP CLIENT
12
HTTP SERVER
HTTP REQUEST X'
CLIENT COMMAND A
CLIENT COMMAND B
RESPONSE TO SERVER COMMAND M
HTTP RESPONSE X'
SERVER COMMAND C
RESPONSE TO CLIENT COMMAND A
HTTP REQUEST Y'
CLIENT COMMAND D
CLIENT COMMAND E
RESPONSE TO SERVER COMMAND C
HTTP RESPONSE Y'
SERVER COMMAND F
SERVER COMMAND G
RESPONSE TO CLIENT COMMAND B
RESPONSE TO CLIENT COMMAND D
RESPONSE TO CLIENT COMMAND E

[Fig. 7]



[Fig. 8]



[Fig. 9]

CLIENT COMMAND SHEET

COMMAND ID
METHOD NAME (e.g., TROUBLE NOTIFICATION)
INPUT PARAMETER (e.g., TROUBLE CONTENT)
STATUS (INITIAL VALUE: NOT SENT)
CLIENT COMMAND EXECUTION RESULT NOTIFYING DESTINATION
OUTPUT PARAMETER (BLANK UNTIL RESPONSE IS ACQUIRED)

[Fig. 10]

SERVER COMMAND SHEET

COMMAND ID
METHOD NAME (e.g., COUNTER ACQUISITION)
INPUT PARAMETER
STATUS (INITIAL VALUE: NOT SENT)
OUTPUT PARAMETER
(BLANK UNTIL PROCESS IS COMPLETED)
SERVER COMMAND NOTIFYING DESTINATION

[Fig. 11]

HTTP REQUEST

```
POST /eaa HTTP/1.1
Content-Type:multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
SOAPAction:"SOAP Action URI"
X-SOAP-Type:Requset

<s:Envelope>
  <--SOAP Request-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response

<s:Envelope>
  <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response

<s:Envelope>
  <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response

<s:Envelope>
  <--SOAP Response-->
</s:Envelope>

--MINE_boundary--
```

PART 1
PART 2
PART 3
PART 4

[Fig. 12]

HTTP RESPONSE

```
HTTP/1.1  200 OK
Content-Type:multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
SOAPAction:"SOAP Action URI"
X-SOAP-Type:Requset

<s:Envelope>
   <--SOAP Request-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response

<s:Envelope>
   <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response

<s:Envelope>
   <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=UTF-8
Content-Transfer-Encoding:8bit
X-SOAP-Type:Response

<s:Envelope>
   <--SOAP Response-->
</s:Envelope>

--MINE_boundary--
```

PART 1

PART 2

PART 3

PART 4

[Fig. 13]

EXEMPLARY PART
DESCRIBING CLIENT COMMAND

```
Content-Type:  text/xml; charset=UTF-8
Content-Transfer Encoding:  8bit
X-SOAP-Type : Request
SOAPAction: "http://www.foo.com/ server/ errorNotification"

<s:Envelope
 xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:n="http://www.foo.com/header"
 xmlns:ns="http://www.foo.com/server"
 s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">


 <s:Header>
    <n:Request ID> 12345 </n:Request ID>
 </s:Header>
  <s:Body>
   <ns:Trouble Notification>
      <ErrorID> 1111 </ErrorID>
      <Description>Hard Disk Drive Trouble </Description>
   </ns:Trouble Notification>
  </s:Body>
</s:Envelope>
```

[Fig. 14]

EXEMPLARY PART DESCRIBING
RESPONSE TO CLIENT COMMAND

```
Content-Type:  text/xml; charset=UTF-8
Content-Transfer-Encoding:  8bit
X-SOAP-Type : Response


<s:Envelope
 xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:n="http://www.foo.com/header"
 xmlns:ns="http://www.foo.com/server"
 s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">


 <s:Header>
    <n:Request ID> 12345 </n:Request ID>
 </s:Header>
  <s:Body>
   <ns:Trouble Notification Response>
      <Reception Result> OK </Reception Result>
   </ns:Trouble Notification Response>
  </s:Body>
</s:Envelope>
```

[Fig. 15]

EXEMPLARY PART
DESCRIBING SERVER COMMAND

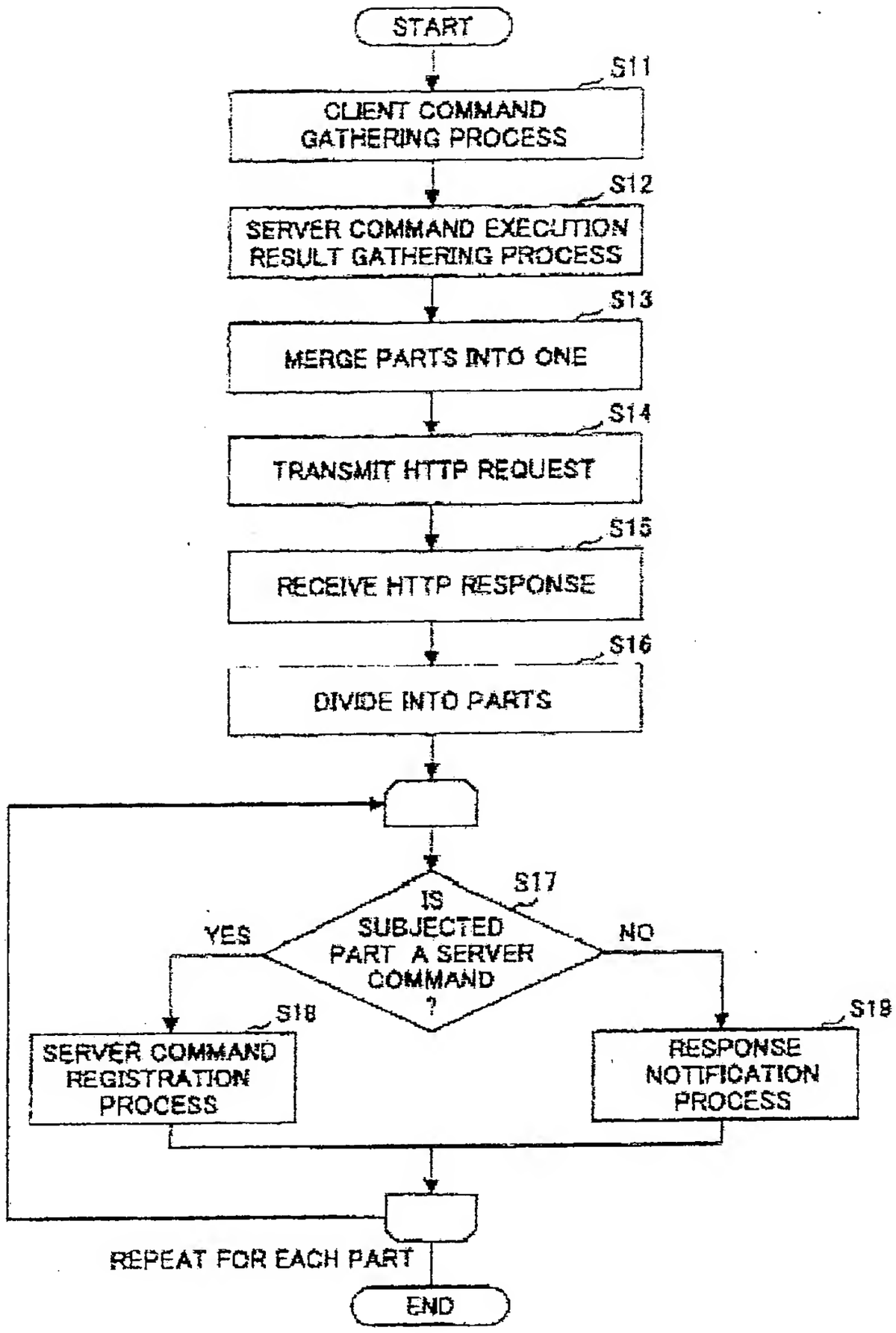
```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type : Request
SOAPAction: "http://www.foo.com/ client/getTemperature"

<s:Envelope
 xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:n="http://www.foo.com/header"
 xmlns:nc="http://www.foo.com/client"
 s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">


 <s:Header>
   <n:Request ID> 98765 </n:Request ID>
 </s:Header>
  <s:Body>
   <nc:Temperature Sensor Value Acqisition>
      <SensorID> 3 </SensorID>
   </nc:Temperature Sensor>
 </s:Body>
</s:Envelope>
```

[Fig. 16]

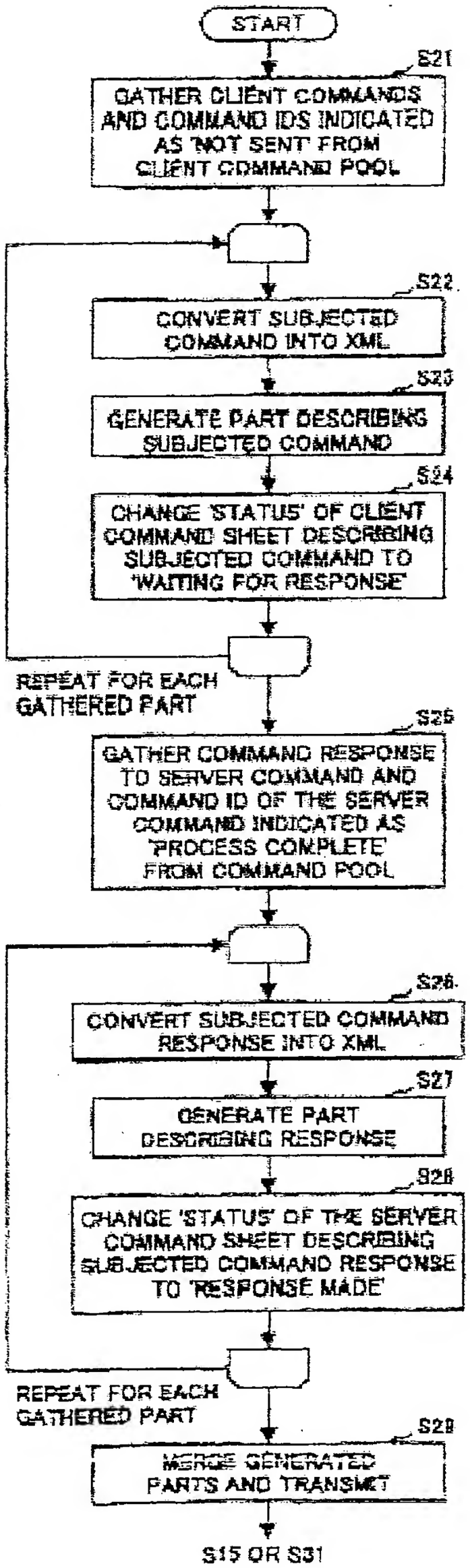
EXEMPLARY PART DESCRIBING
RESPONSE TO SERVER COMMAND

```
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
X-SOAP-Type : Response


<s:Envelope
 xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:se="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:n="http://www.foo.com/header"
 xmlns:nc="http://www.foo.com/client"
 s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">


 <s:Header>
   <n:Request ID> 98765 </n:Request ID>
 </s:Header>
  <s:Body>
   <nc:Temperature Sensor Value Acqisition Response>
      <Temperature> 52 </Temperature>
   </nc:Temperature Sensor Value Acqisition Response>
 </s:Body>
</s:Envelope>
```

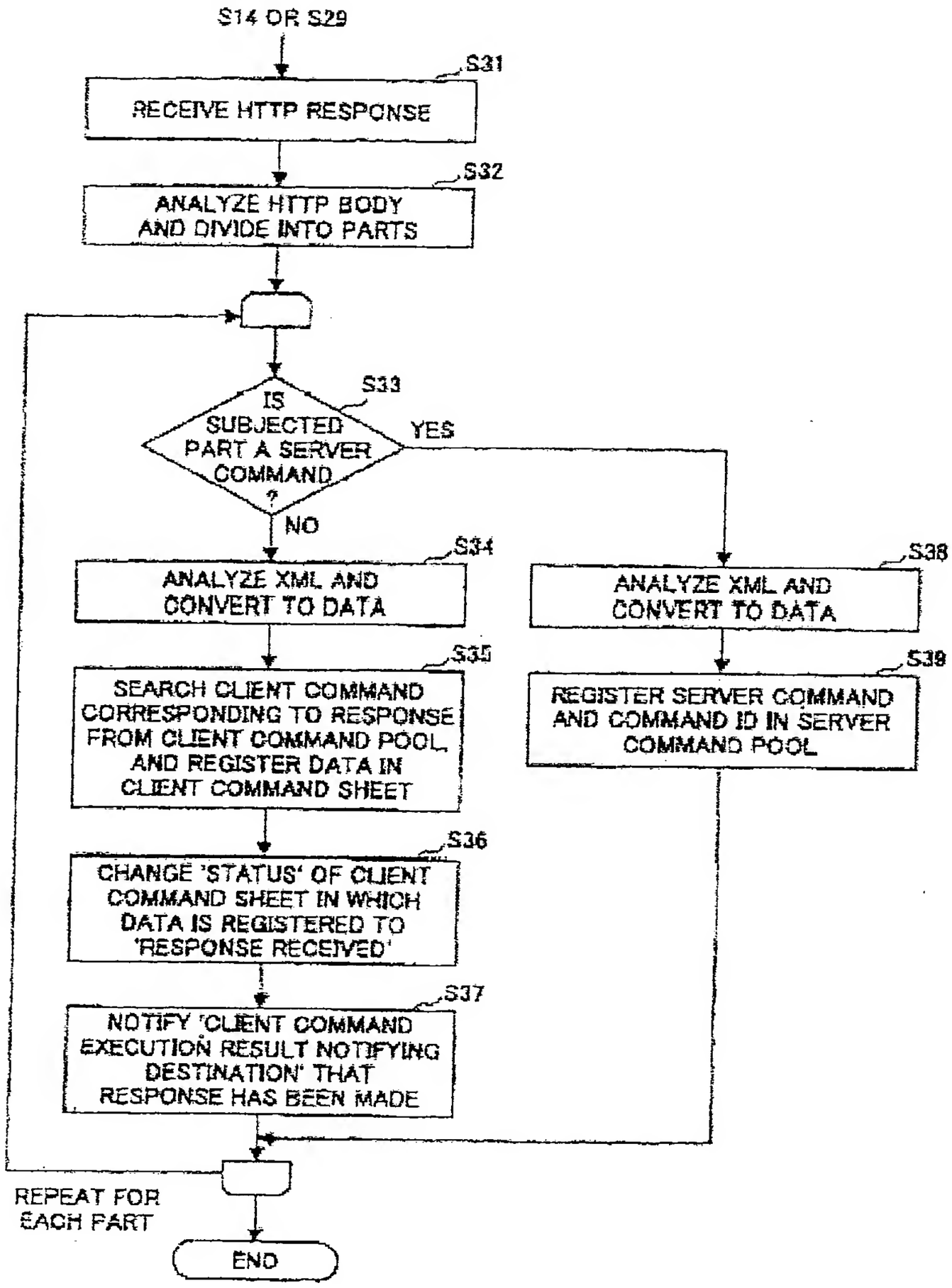
[Fig. 17]


START
S11
CLIENT COMMAND GATHERING PROCESS
S12
SERVER COMMAND EXECUTION RESULT GATHERING PROCESS
S13
MERGE PARTS INTO ONE
S14
TRANSMIT HTTP REQUEST
S15
RECEIVE HTTP RESPONSE
S16
DIVIDE INTO PARTS
S17
IS SUBJECTED PART A SERVER COMMAND ?
YES
NO
S18
SERVER COMMAND REGISTRATION PROCESS
S19
RESPONSE NOTIFICATION PROCESS
REPEAT FOR EACH PART
END

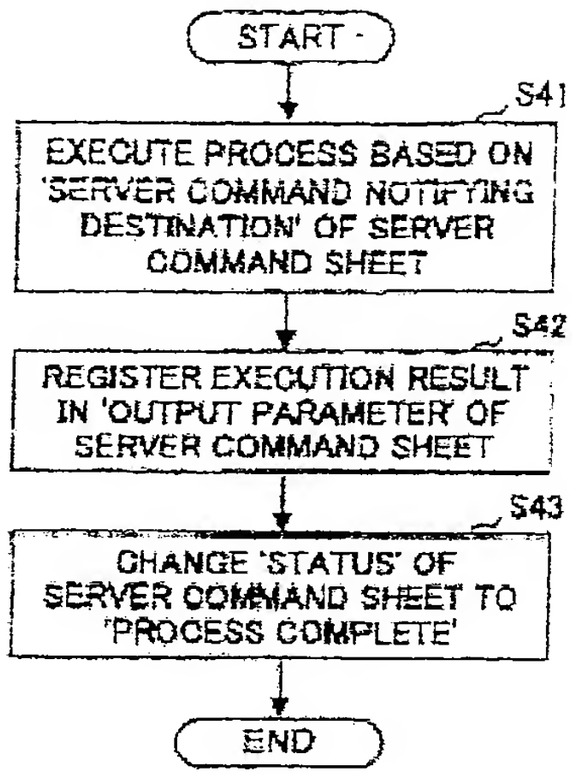
[Fig. 18]


START
S21
GATHER CLIENT COMMANDS AND COMMAND IDS INDICATED AS NOT SENT FROM CLIENT COMMAND POOL
S22
CONVERT SUBJECTED COMMAND INTO XML
S23
GENERATE PART DESCRIBING SUBJECTED COMMAND
S24
CHANGE 'STATUS' OF CLIENT COMMAND SHEET DESCRIBING SUBJECTED COMMAND TO 'WAITING FOR RESPONSE'
REPEAT FOR EACH GATHERED PART
S25
GATHER COMMAND RESPONSE TO SERVER COMMAND AND COMMAND ID OF THE SERVER COMMAND INDICATED AS 'PROCESS COMPLETE' FROM COMMAND POOL
S26
CONVERT SUBJECTED COMMAND RESPONSE INTO XML
S27
GENERATE PART DESCRIBING RESPONSE
S28
CHANGE 'STATUS' OF THE SERVER COMMAND SHEET DESCRIBING SUBJECTED COMMAND RESPONSE TO 'RESPONSE MADE'
REPEAT FOR EACH GATHERED PART
S29
MERGE GENERATED PARTS AND TRANSMIT
S15 OR S31

[Fig. 19]


S14 OR S29
S31
RECEIVE HTTP RESPONSE
S32
ANALYZE HTTP BODY AND DIVIDE INTO PARTS
S33
IS SUBJECTED PART A SERVER COMMAND ?
YES
NO
S34
ANALYZE XML AND CONVERT TO DATA
S35
SEARCH CLIENT COMMAND CORRESPONDING TO RESPONSE FROM CLIENT COMMAND POOL AND REGISTER DATA IN CLIENT COMMAND SHEET
S36
CHANGE 'STATUS' OF CLIENT COMMAND SHEET IN WHICH DATA IS REGISTERED TO 'RESPONSE RECEIVED'
S37
NOTIFY 'CLIENT COMMAND EXECUTION RESULT NOTIFYING DESTINATION' THAT RESPONSE HAS BEEN MADE
S38
ANALYZE XML AND CONVERT TO DATA
S39
REGISTER SERVER COMMAND AND COMMAND ID IN SERVER COMMAND POOL
REPEAT FOR EACH PART
END

[Fig. 20]


START
S41
EXECUTE PROCESS BASED ON 'SERVER COMMAND NOTIFYING DESTINATION' OF SERVER COMMAND SHEET
S42
REGISTER EXECUTION RESULT IN 'OUTPUT PARAMETER' OF SERVER COMMAND SHEET
S43
CHANGE 'STATUS' OF SERVER COMMAND SHEET TO 'PROCESS COMPLETE'
END

[Fig. 21]

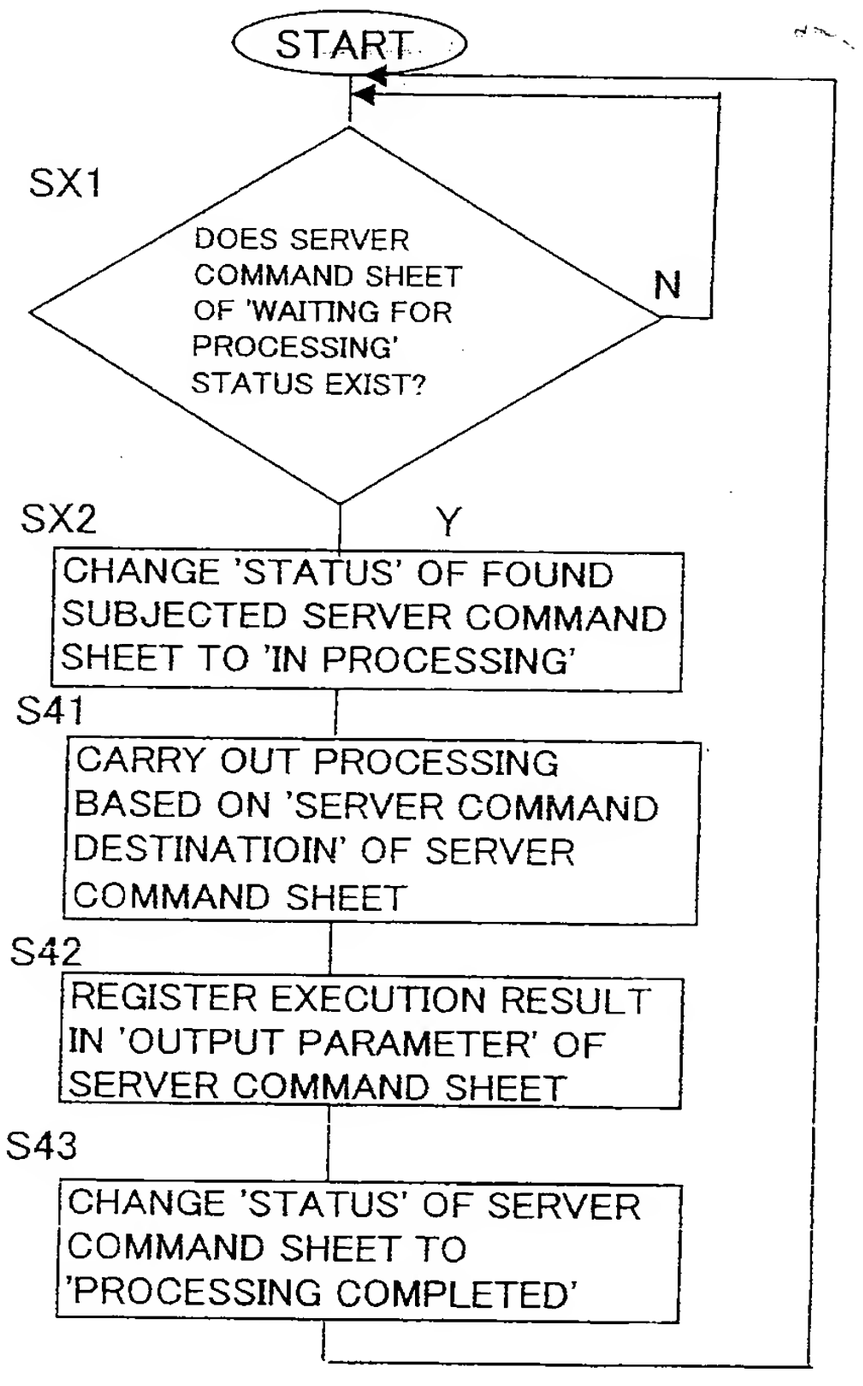

START
SX1
DOES SERVER COMMAND SHEET OF 'WAITING FOR PROCESSING' STATUS EXIST?
N
Y
SX2
CHANGE 'STATUS' OF FOUND SUBJECTED SERVER COMMAND SHEET TO 'IN PROCESSING'
S41
CARRY OUT PROCESSING BASED ON 'SERVER COMMAND DESTINATIOIN' OF SERVER COMMAND SHEET
S42
REGISTER EXECUTION RESULT IN 'OUTPUT PARAMETER' OF SERVER COMMAND SHEET
S43
CHANGE 'STATUS' OF SERVER COMMAND SHEET TO 'PROCESSING COMPLETED'

Fig. 22

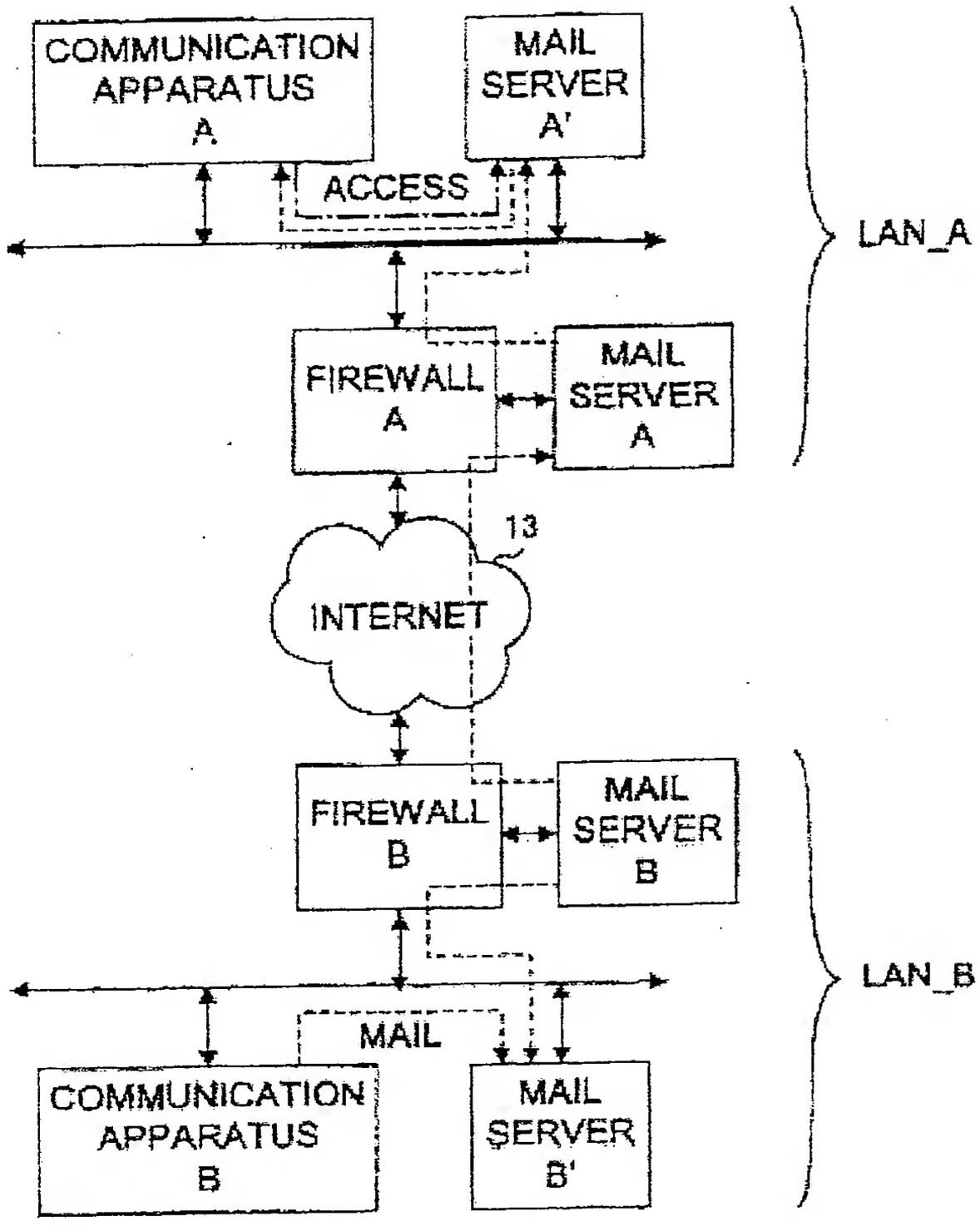
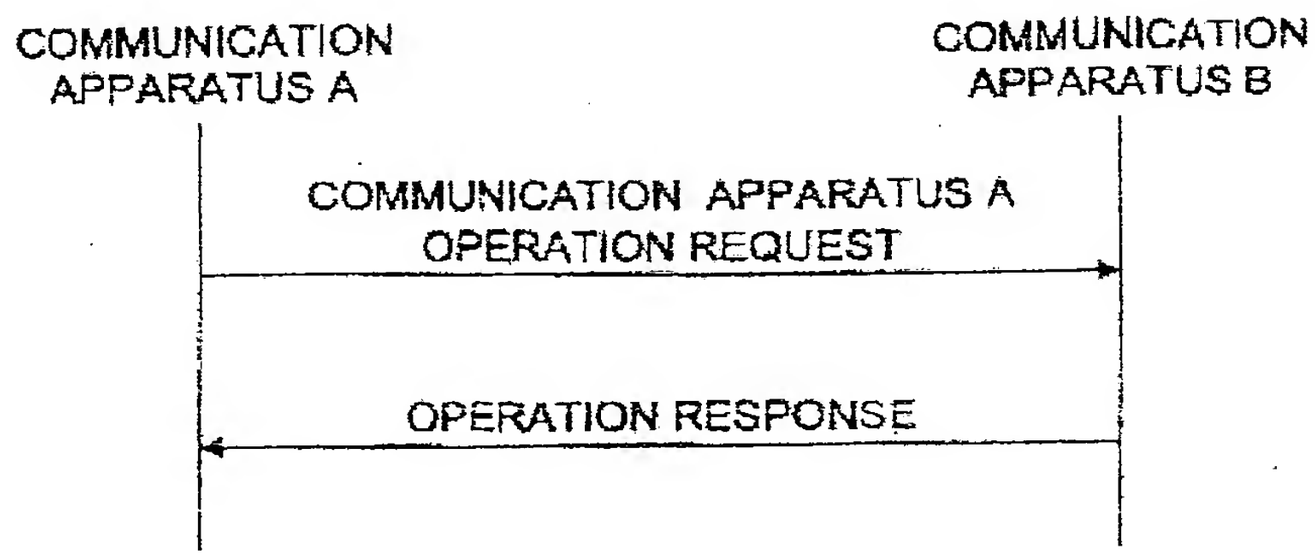


Fig. 23

(A)



(B)

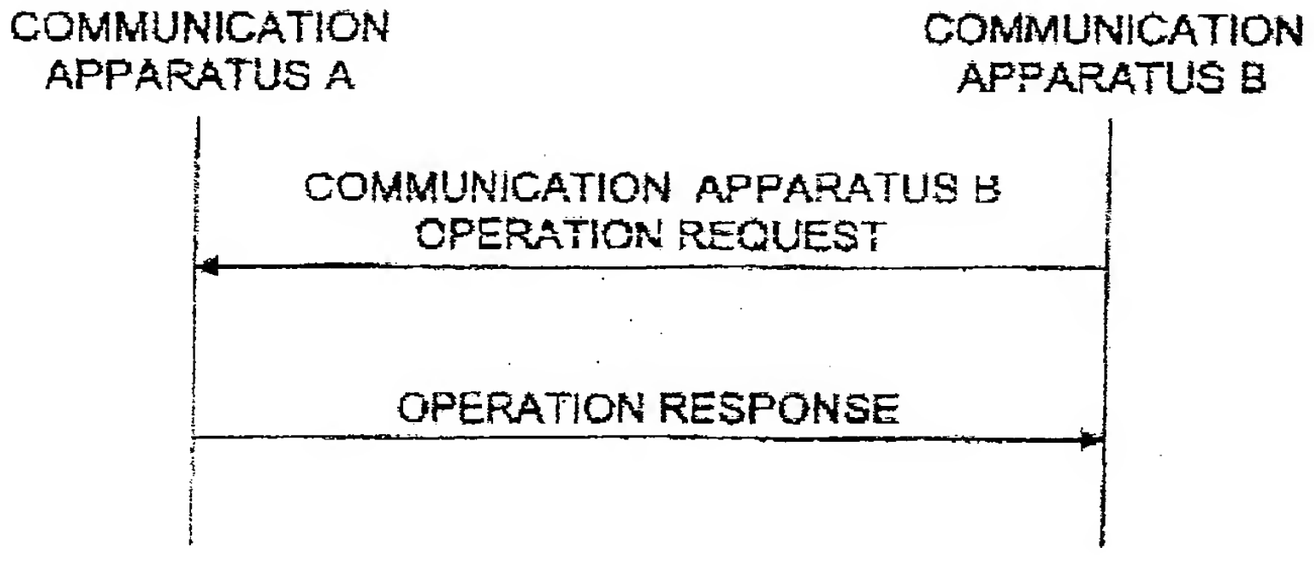

COMMUNICATION APPARATUS A
COMMUNICATION APPARATUS B
COMMUNICATION APPARATUS B OPERATION REQUEST
OPERATION RESPONSE

Fig. 24

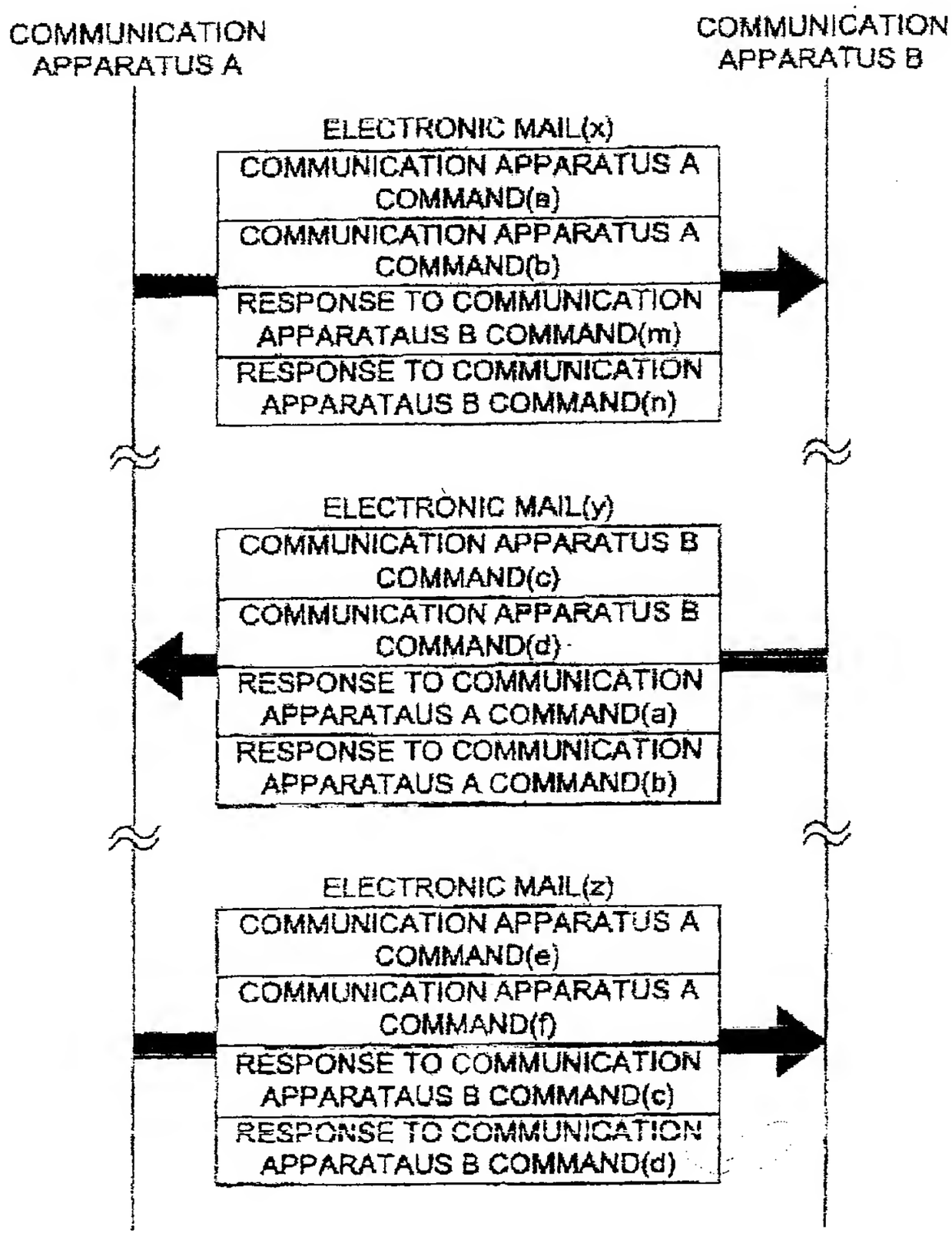

COMMUNICATION APPARATUS A
COMMUNICATION APPARATUS B
ELECTRONIC MAIL(x)
COMMUNICATION APPARATUS A COMMAND(a)
COMMUNICATION APPARATUS A COMMAND(b)
RESPONSE TO COMMUNICATION APPARATAUS B COMMAND(m)
RESPONSE TO COMMUNICATION APPARATAUS B COMMAND(n)
ELECTRONIC MAIL(y)
COMMUNICATION APPARATUS B COMMAND(c)
COMMUNICATION APPARATUS B COMMAND(d)
RESPONSE TO COMMUNICATION APPARATAUS A COMMAND(a)
RESPONSE TO COMMUNICATION APPARATAUS A COMMAND(b)
ELECTRONIC MAIL(z)
COMMUNICATION APPARATUS A COMMAND(e)
COMMUNICATION APPARATUS A COMMAND(f)
RESPONSE TO COMMUNICATION APPARATAUS B COMMAND(c)
RESPONSE TO COMMUNICATION APPARATAUS B COMMAND(d)

Fig. 25

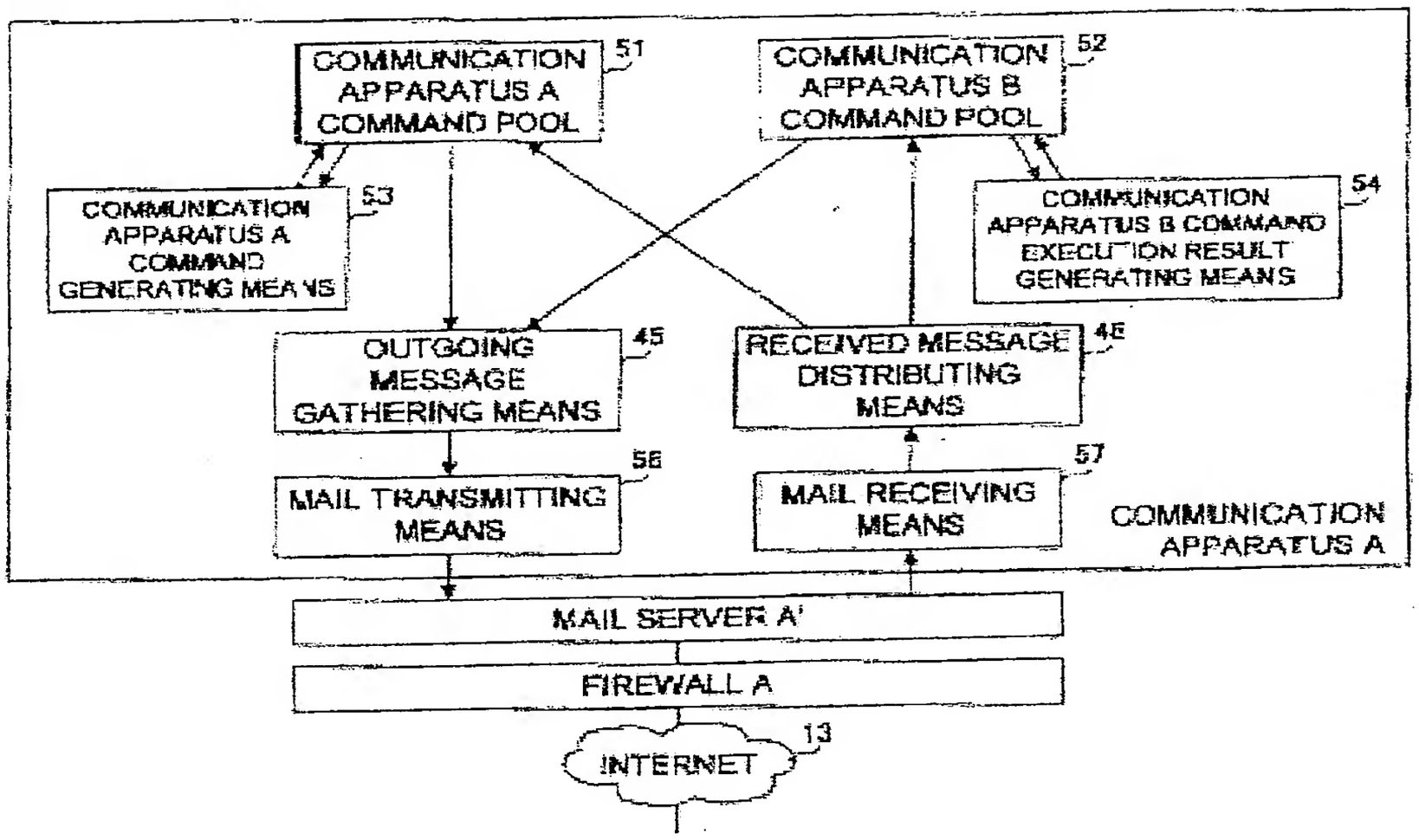

COMMUNICATION APPARATUS A COMMAND POOL
51
COMMUNICATION APPARATUS B COMMAND POOL
52
COMMUNICATION APPARATUS A COMMAND GENERATING MEANS
53
COMMUNICATION APPARATUS B COMMAND EXECUTION RESULT GENERATING MEANS
54
OUTGOING MESSAGE GATHERING MEANS
55
RECEIVED MESSAGE DISTRIBUTING MEANS
56
MAIL TRANSMITTING MEANS
57
MAIL RECEIVING MEANS
COMMUNICATION APPARATUS A
MAIL SERVER A'
FIREWALL A
INTERNET
13

Fig. 26

```
From: deviceA@foo.com
To: deviceB@bar.com
Subject: Multi Message #0000001
Date: Web, 30 Jul 2003 10:00:00  +0900
Content-Type:multipart/mixed;boundary=MIME_boundary
Content-Length:nnnn

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
SOAPAction:"SOAP Action URI"
X-SOAP-Type:Requset

<s:Envelope>
   <--SOAP Request-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
X-SOAP-Type:Response

<s:Envelope>
   <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
X-SOAP-Type:Response

<s:Envelope>
   <--SOAP Response-->
</s:Envelope>

--MIME_boundary
Content-Type:text/xml;charset=iso-2022-jp
Content-Transfer-Encoding:7bit
X-SOAP-Type:Response

<s:Envelope>
   <--SOAP Response-->
</s:Envelope>

--MINE_boundary--
```

PART 1

PART 2

PART 3

PART 4

Fig. 27

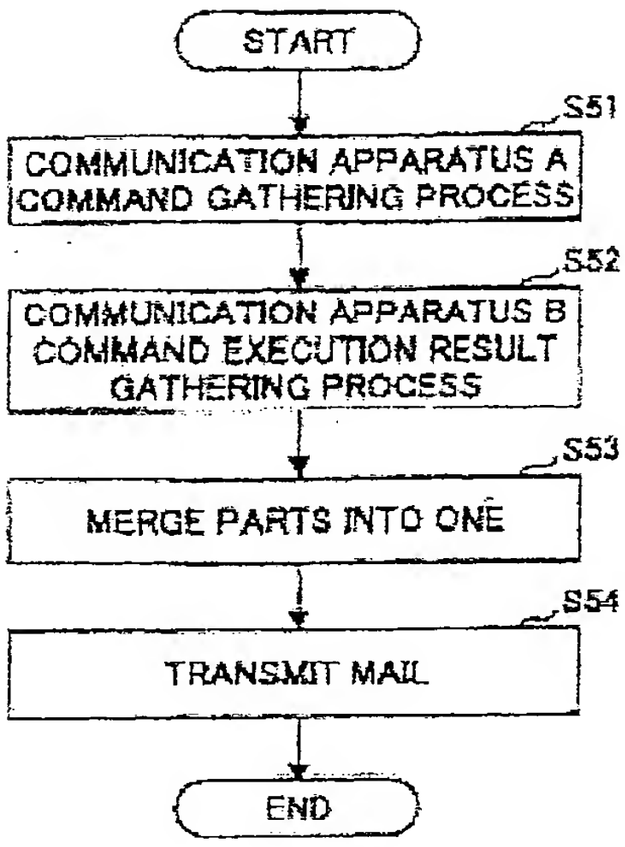

START
S51
COMMUNICATION APPARATUS A COMMAND GATHERING PROCESS
S52
COMMUNICATION APPARATUS B COMMAND EXECUTION RESULT GATHERING PROCESS
S53
MERGE PARTS INTO ONE
S54
TRANSMIT MAIL
END

Fig. 28

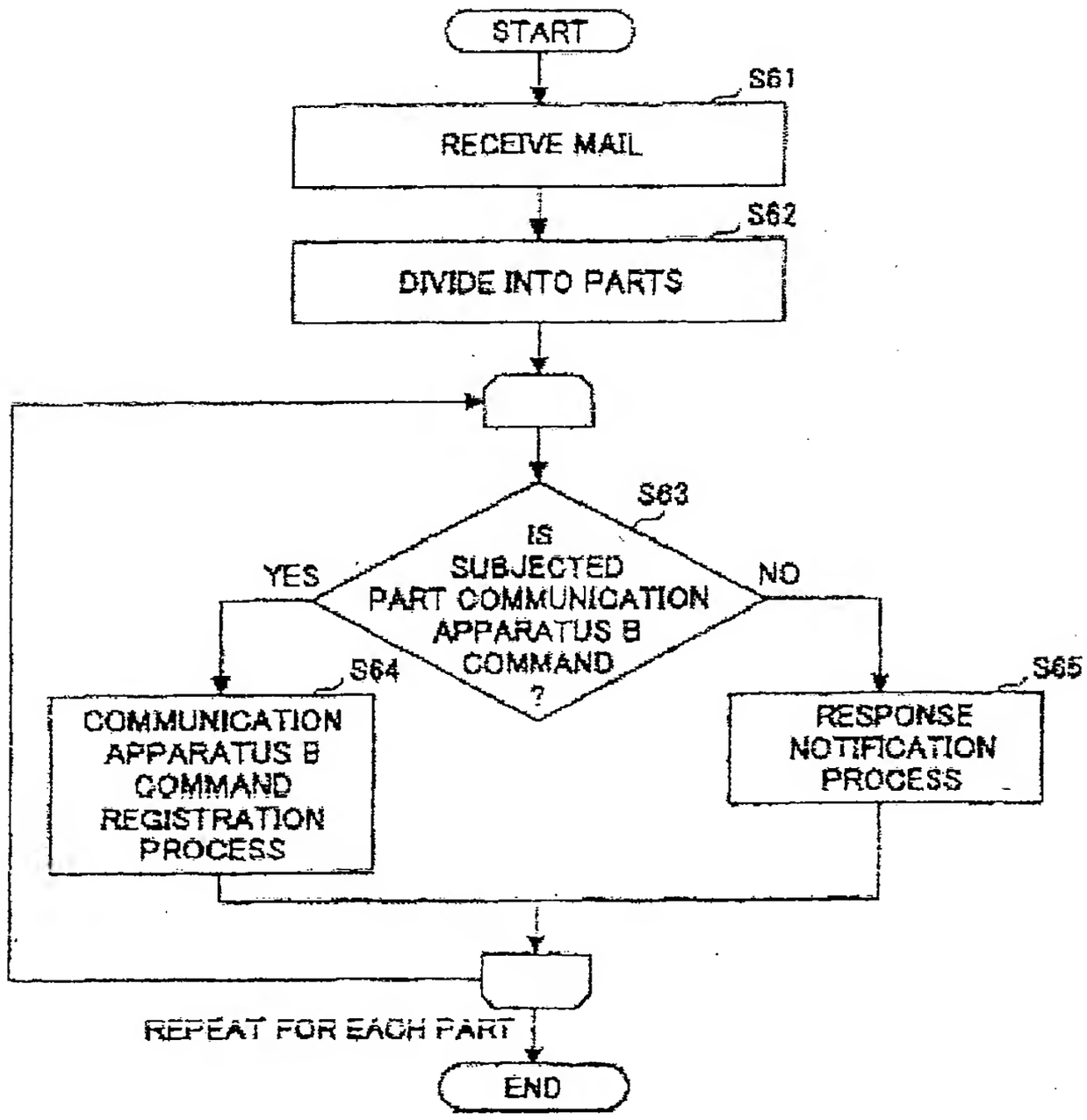
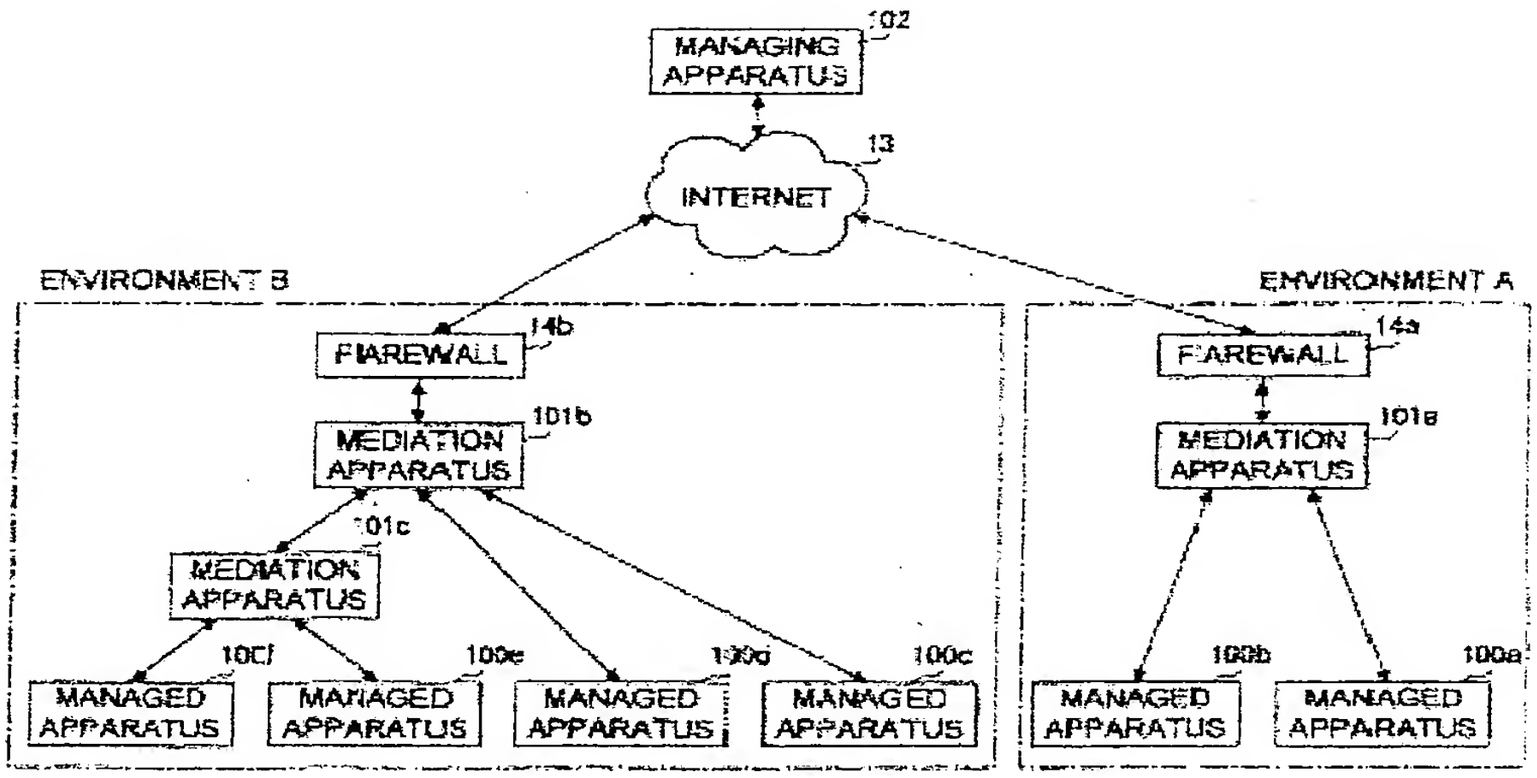

START
S61
RECEIVE MAIL
S62
DIVIDE INTO PARTS
S63
IS SUBJECTED PART COMMUNICATION APPARATUS B COMMAND ?
YES
NO
S64
COMMUNICATION APPARATUS B COMMAND REGISTRATION PROCESS
S65
RESPONSE NOTIFICATION PROCESS
REPEAT FOR EACH PART
END

Fig. 29


MANAGING APPARATUS
102
INTERNET
13
ENVIRONMENT B
ENVIRONMENT A
FIREWALL
14b
FIREWALL
14a
MEDIATION APPARATUS
101b
MEDIATION APPARATUS
101a
101c
MEDIATION APPARATUS
100d
100c
MANAGED APPARATUS
MANAGED APPARATUS
MANAGED APPARATUS
MANAGED APPARATUS
100b
100a
MANAGED APPARATUS
MANAGED APPARATUS